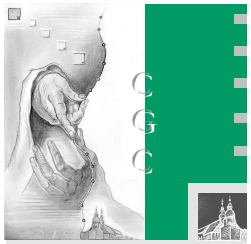


Werkzeuge der Informatik

Einführung in Unix/Linux

G. Zachmann

Clausthal University, Germany

zach@in.tu-clausthal.de

Literatur

- Bücher über Unix gibt es wie Sand am Meer ...
- Z.B.:
  - Jerry Peek, Tim O'Reilly & Mike Loukides: *UNIX Power Tools*. O'Reilly & Associates.
  - Michael Kofler: *Linux - Installation, Konfiguration, Anwendung*. Addison-Wesley.
  - Daniel J. Barrett: *Linux kurz und gut*. O'Reilly, September 2004.
- Parallel bzw. ergänzend dazu Online-Literatur auf der Web-Seite! (und noch viel mehr im Netz)

Weiterführender Kurs

- Vom GWDG in Göttingen:
  - Videoaufzeichnungen der letzten Veranstaltung
  - Siehe: http://www.uni-math.gwdg.de/linuxuebung/
  - Schon recht fortgeschritten
- Manchmal auch in unserem Rechenzentrum:
  - Termine siehe www.rz.tu-clausthal.de

Heimarbeit ... (wer hat kein Linux?)

- Irgendeine Distro kaufen oder vom RZ beziehen und installieren:
  - Z.B.: Kubuntu, Mandrake, ...
  - http://ftp.tu-clausthal.de/ftp/linux/ oder ftp://ftp.tu-clausthal.de/pub/linux/
- Knoppix-CD vom RZ ziehen:
  - Keine Installation nötig
  - ftp://ftp.tu-clausthal.de/pub/linux/knoppix/
  - Achtung: Files sichern vor dem Abschalten!
- In beiden Fällen: ISO ziehen und CD brennen
- Cygwin
  - www.cygwin.com
  - Achtung: Execs laufen nicht auf den Linux-PCs im Pool

... und remote an der Uni

- Account am IfI:
  - Jede Gruppe bekommt in der ersten Übung einen generischen Account
  - Die Accounts werden nach dem Semester gelöscht (Daten extern sichern!)
- Remote einloggen auf kaosus

```
ssh login.in.tu-clausthal.de -l account
```

- Daten hin- und herkopieren

```
scp source-dir account@login.in.tu-
clausthal.de:/home/account/...
```

Remote Login

- Ist oft sehr praktisch! Z.B., wenn ...
  - ... auf dem aktuellen Rechner bestimmte Software nicht installiert ist
  - ... man einen anderen Rechner administrieren muß
    - z.B. Web-Seiten von auswärts editieren

Wer braucht UNIX?

> *"Unix ist zwar ein Mainframe-Betriebssystem (und damit obsolet) hat aber noch viele Anhänger."*
> Windows MSCE-Training-Guide Windows 2000 Server
> Kapitel 2.6.3 "Zusammenspiel mit UNIX", Verlag Markt & Technik

- Programmierer
- Web-Server
- Distributed Computing
- Wer braucht UNIX *nicht* (unbedingt) ?
  - Sekretärinnen
  - Büro- und Business-Software (Word, Buchhaltung, Powerpoint, Lagerhaltung, ...)

Vorteile von UNIX

- Extrem ausgereift (besonders die kommerziellen Unices)
- Gut durchdachtes Konzept von Anfang an
  - "Alles ist ein File"
  - "Alles ist ein Prozeß"
- Von Anfang an Multi-User- und Multi-Task-fähig
- Relativ sicher
- Flexibler
- Performanter
- Wesentlich leichter zu administrieren (wenn die Lernkurve erst einmal durchschritten ist)
- Auf allen Plattformen verfügbar

Plattformen

- Sun (Solaris)
- HP (HP-UX)
- SGI (IRIX)
- IBM (AIX)
- Mac (OS-X)
- PC (Linux)
- PDA
- Set-top boxes
- Armbanduhr
- Auto
- ...

http://www.research.ibm.com/WearableComputing/index.html

http://www.linuxdevices.com/

Die Erfinder

- Ca. 1970:
  - Haben UNIX und C erfunden!

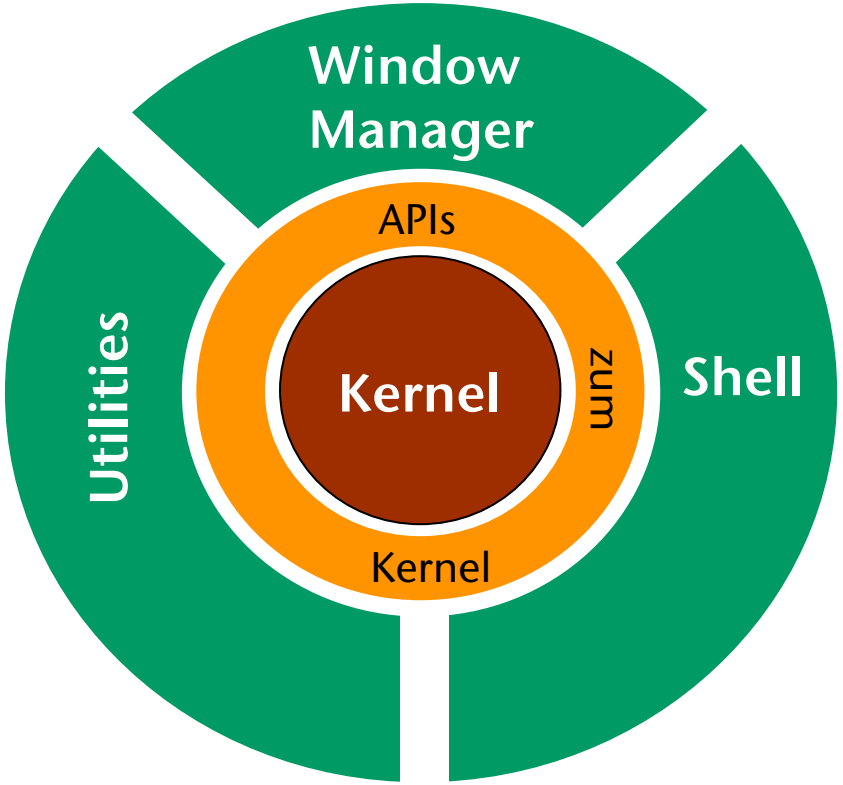
Ken Thompson and Dennis Ritchie
Your new heroes

Was ist UNIX?

- Ein Betriebssystem
- Eine Sammlung von nützlichen Tools
- Eine (Computer-)Kultur

UNIX Komponenten

- Kernel: Herz des OS, managt Hardware & Programme
- Shell: eine Applikation, nimmt Kommandos entgegen und führt sie aus (CLI)
- Utilities: viele kleine (und große) Tools zur täglichen Arbeit, z.B. Files kopieren, ASCII-Texte editieren, ...

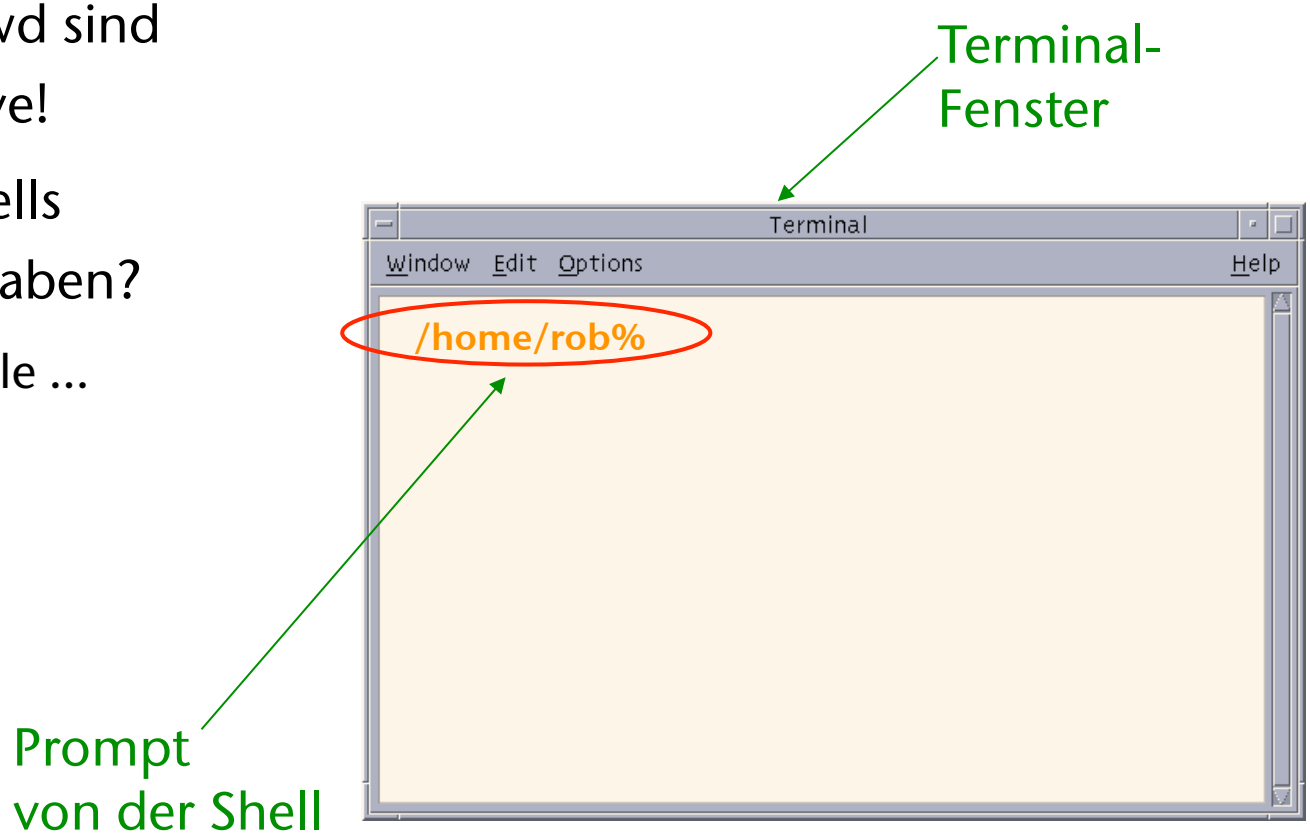


Deutsches UNIX

- Große Unsitte
  - Wegen Terminologie
- Also: **Englisch einstellen!**
  - In der tcsh: **setenv LANG en** und **setenv KDE_LANG de**
  - In der bash: **export LANG=en** und **export KDE_LANG=de**
  - Und/oder auf dem Login-Screen Englisch einstellen
  - Oder: KDE Control Center → Regional & Accessibility → Country/ Region & Language

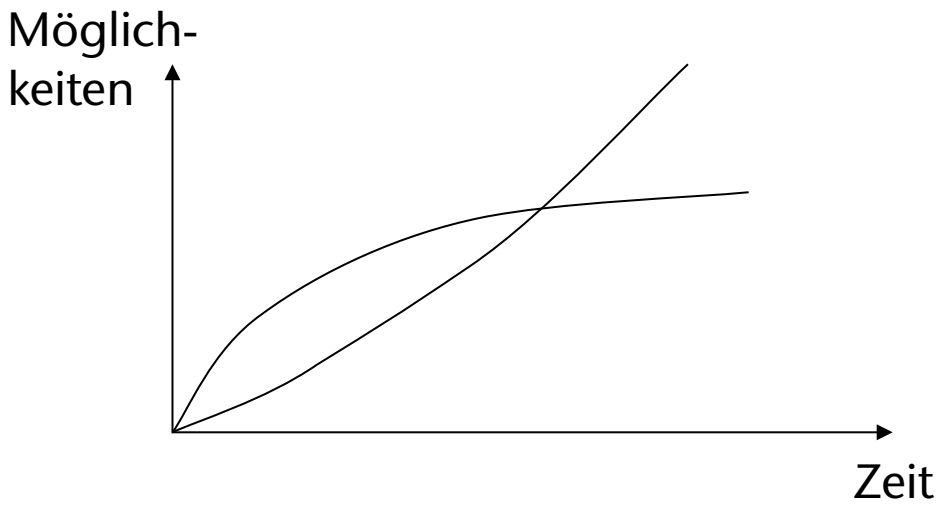
Erstes Einloggen

- Wie bekommt man eine Shell / (Terminal-)Fenster?
  - An der "Konsole" ("console")
  - Remote (ssh, rlogin, telnet)
- Login/passwd sind case-sensitive!
- Wieviele Shells kann man haben?
  - Beliebig viele ...
- Das Prompt

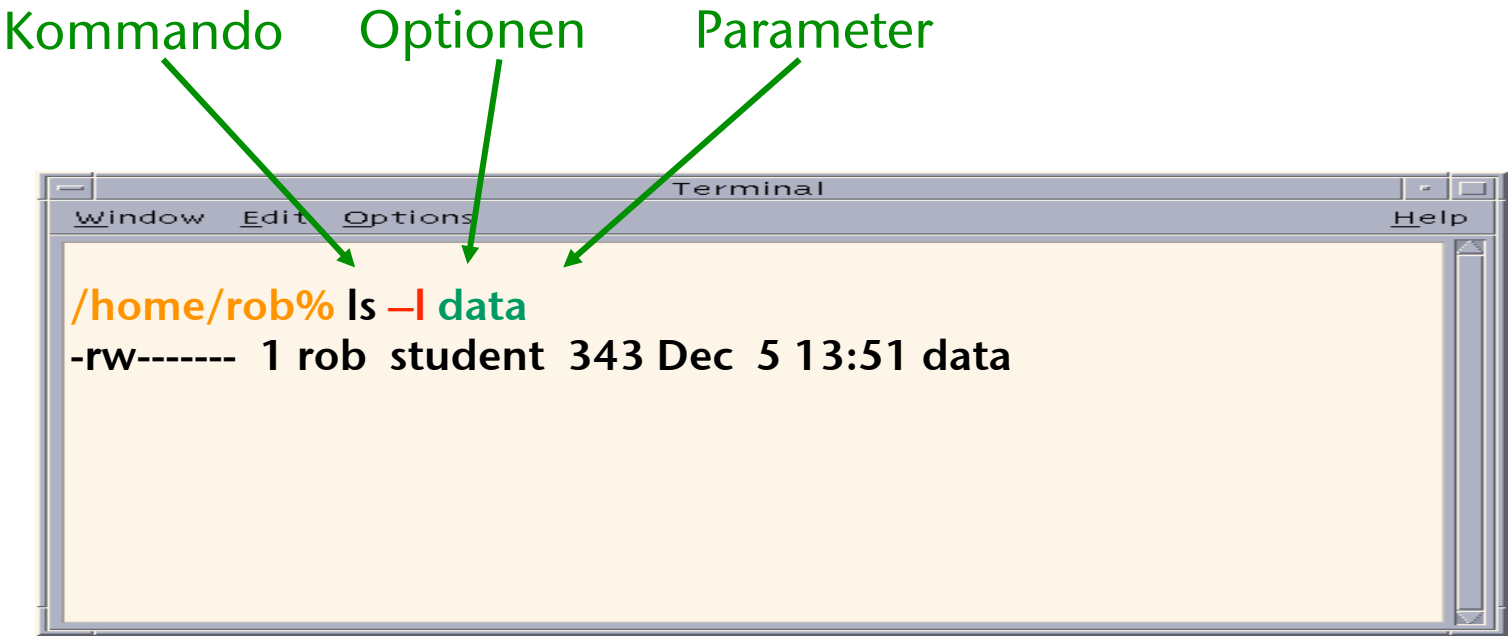


Das User-Interface

- Ist immer noch die Kommandozeile
  (CLI = command line interface)
- Für Programmierer ist CLI sehr viel effizienter!
- Lernkurve ist natürlich länger ("steiler")



Aufbau einer Kommandozeile



- Optionen (options, flags): ändern Verhalten
- Parameter: i.a. Files, auf denen Kommando operiert

Editieren der Kommandozeile

- In der Zeile:

Taste	Funktion
Tab	File- / Command-Completion
Ctrl-B / Ctrl-F	Wortweise vor / zurück springen
Ctrl-W	Voriges Wort löschen
Ctrl-U / Ctrl-K	Zeile bis zum Anfang / Ende löschen
Ctrl-A / Ctrl-E	An Ende / Anfang springen

- In der History:

Taste	Funktion
Cursor-Up / -Down	In der History rauf / runter
Ctrl-P / Ctrl-N	Match in der History nach oben / unten suchen

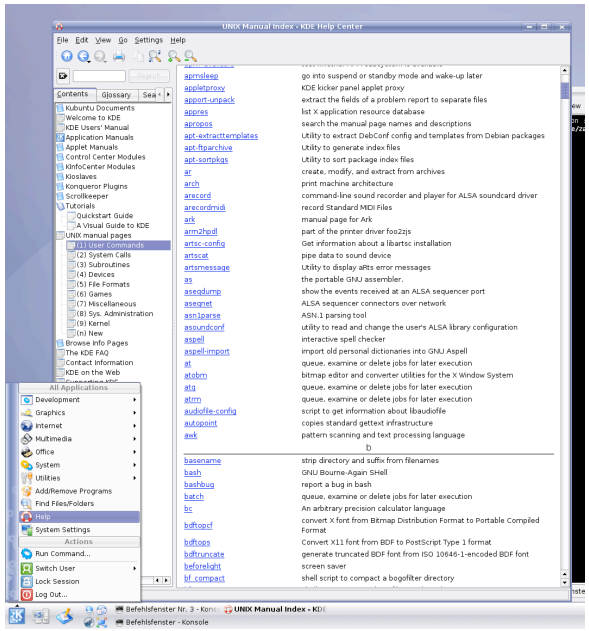
Kommandowiederholung

Komando	Bedeutung
!!	Letztes Kommando wiederholen
!string	Kommando, das mit 'string' beginnt, wiederholen
!17	Kommando mit Nummer 17 i.d. History wiederholen
^a^b	Letztes Kommando wiederholen, dabei das erste Vorkommen von 'a' durch 'b' ersetzen

- History anzeigen: **history** (**alias h**)

More input? Hilfe zur Selbsthilfe

- 4 Arten von Informationsquellen:
  - Man Pages
  - HTML-Seiten
  - Unter KDE: "K"-Menü → Help, dann z.B. "UNIX manual pages"



- Man Pages:
  - **`man cmd`** : zeigt Man-Page zu **`cmd`** an
  - (Programm oder Funktion)
  - **`man -k keyword`** – Alle Man-Pages nach keyword durchsuchen (nur die Titel-Zeile jeder Man-Page)
  - **`man -K keyword`** – Alle Man-Pages nach keyword durchsuchen (komplette Seite)
  - Start-Menü → Help

Format of each man page

Name	Name und 1-zeilige Beschreibung
Syntax	
Description	Ausführliche Beschreibung
Options	
Files	Liste von Files wichtig für diesen Befehl
Return values	
Diagnostics	Mögliche Fehlermeldungen und Ursachen
Bugs	Bekannte Bugs und Unzulänglichkeiten
See also	Verwandte Befehle und Infos

HTML-Seiten

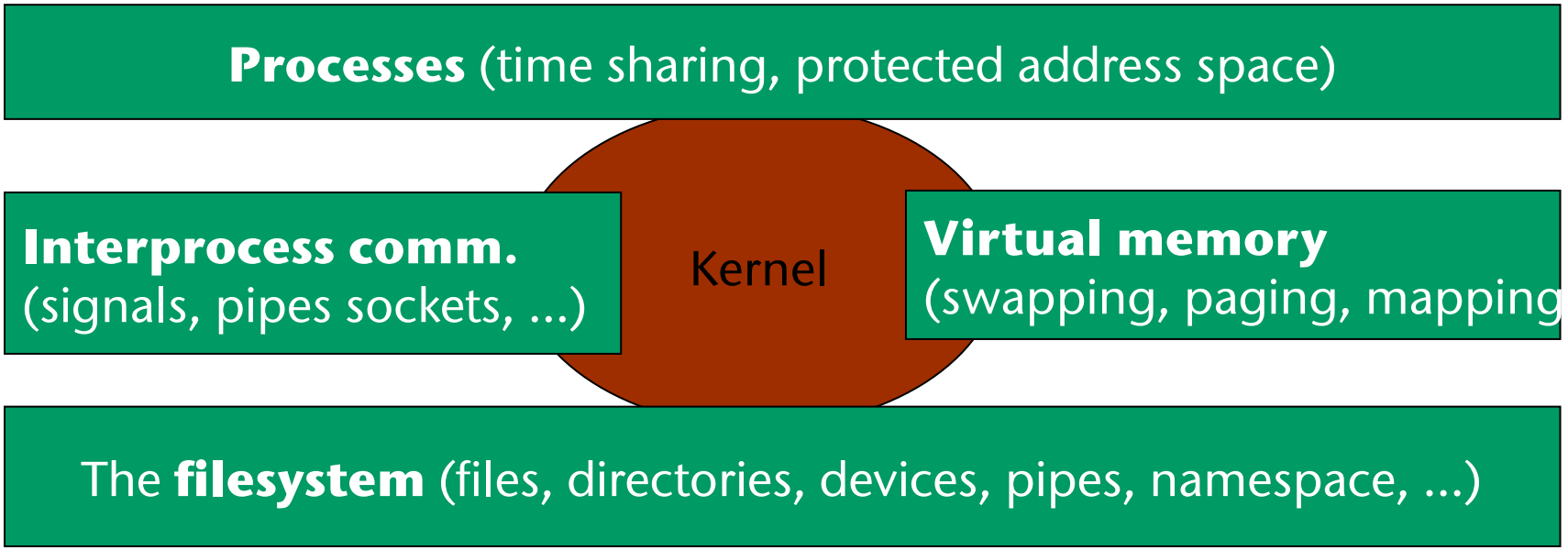
- Hauptproblem: diese zu finden
- Normalerweise in **/usr/share/docs** oder **/usr/local/share/**
- Hilfsmittel: **locate**
- Dann:

Grundregeln unter UNIX

- Don't Panic!
- RTFM! ("read the f*ing manual")
- Probieren geht über studieren ...

UNIX-Konzepte

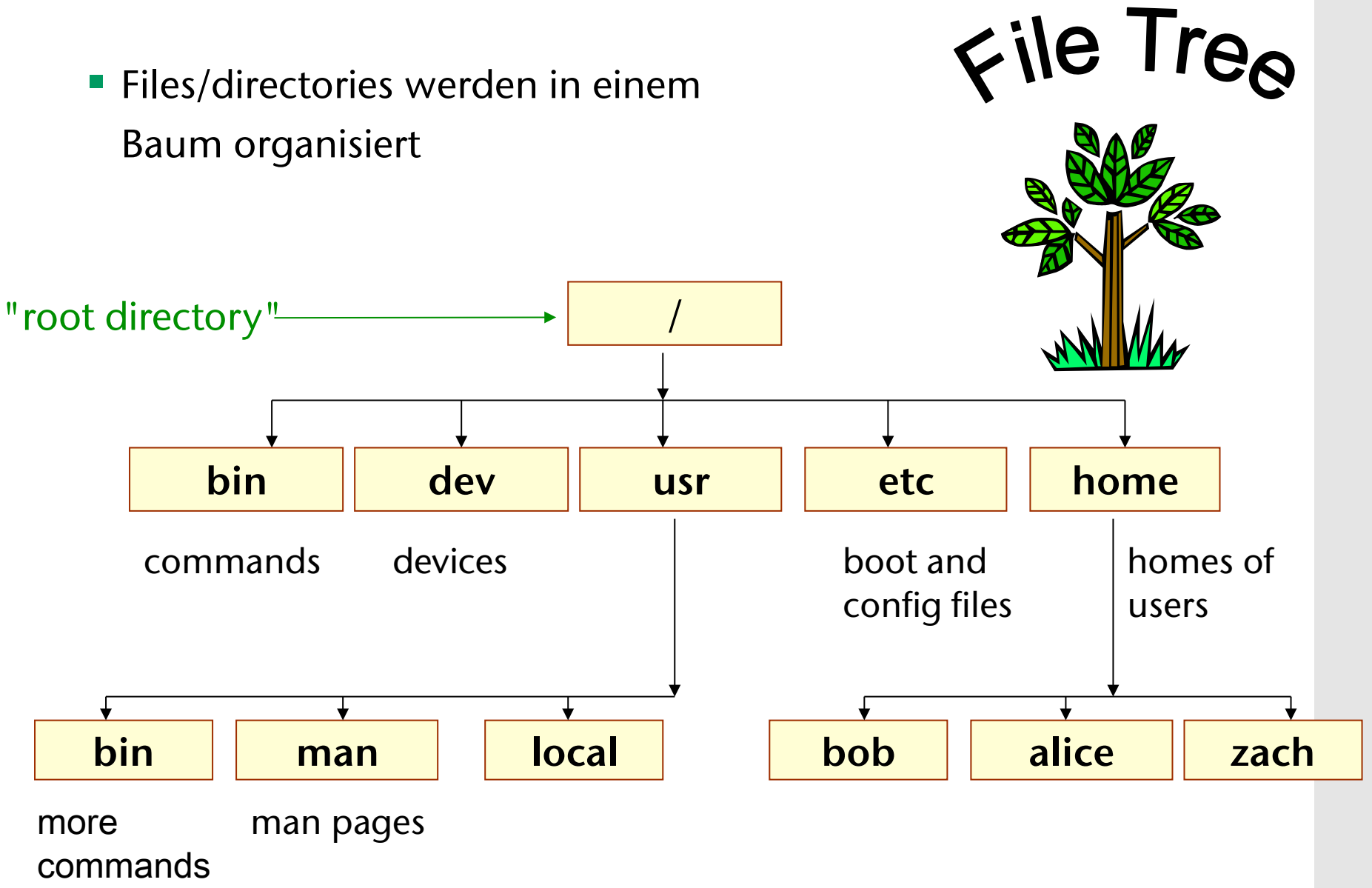
- Einige wenige Grundkonzepte:
  - Alles ist ein File (Programm, Daten, Speicher, ...)
  - Alles ist ein Prozeß (OS, laufendes Programm, Editor, Shell, ...)
  - Viele kleine Utilities, die kombiniert werden können
  - ...



Das Filesystem

- Directories ("Folders") und Files
- File enthält sequentielle Folge von Zeichen (Bytes)
- Interpretation ist Sache des benutzenden Programms:
  - Text, Zahlen, Programm, Speicherauszug, ...
- Jeder File hat einen Namen:
  - Case-sensitive! (UNIX allg.)
  - Länge typ. bis zu 1024
  - Können beliebige Zeichen enthalten –
    besser nur alphanumerische Zeichen und Underscore!
- Directory ("Verzeichnis"):
  - Enthält Name von File und Verweis darauf
  - Spezieller File

- Files/directories werden in einem Baum organisiert

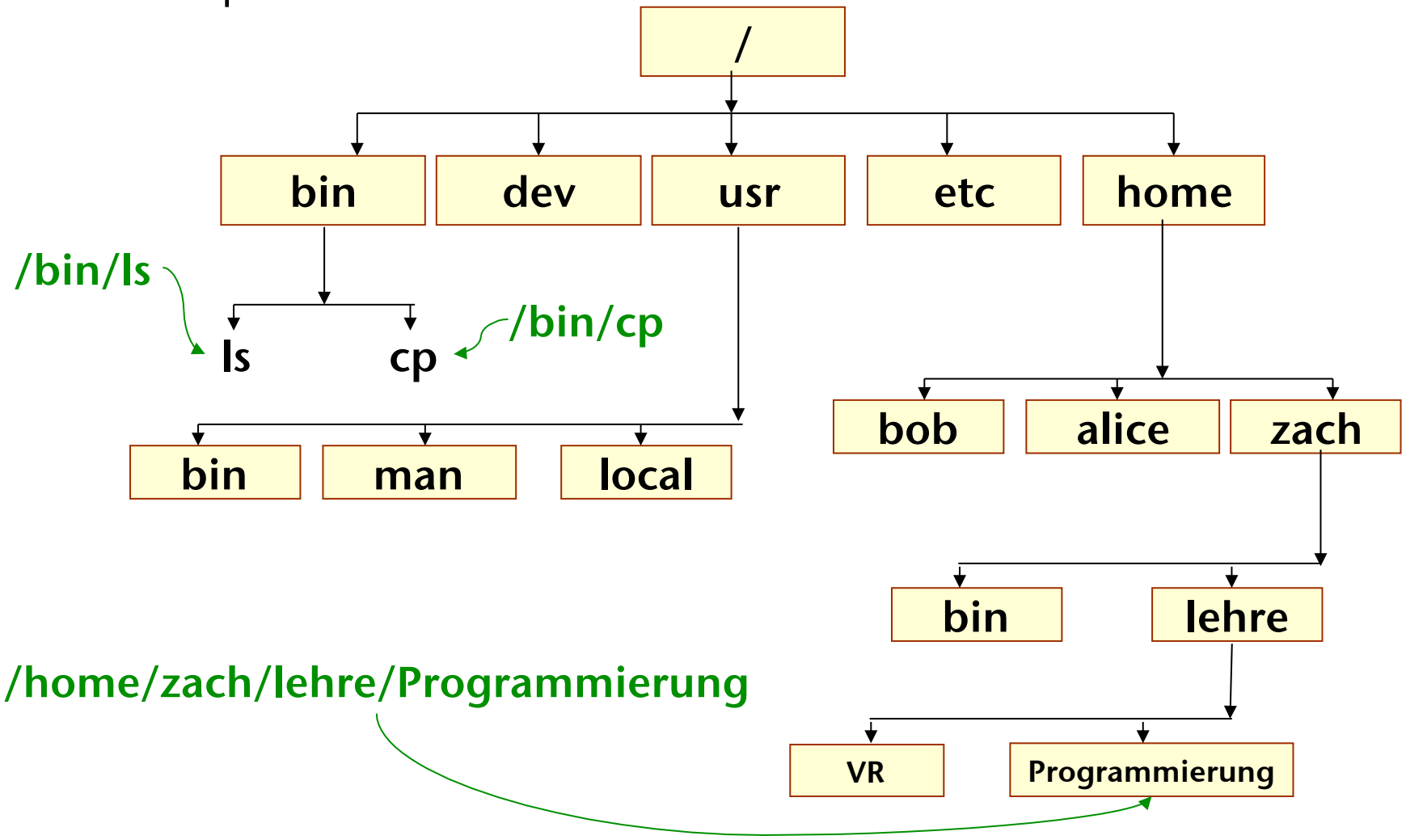


Eindeutigkeit

- Definition "Pfadname" (*pathname*) eines Files:
  Konkatenierung aller Verzeichnisnamen und des Filenamens auf dem Weg von der Wurzel bis zum File, getrennt durch **/**
- Eindeutigkeit:
  - Files im selben Verzeichnis müssen verschiedene Namen haben
  - Files in verschiedenen Directories dürfen gleiche Namen haben!

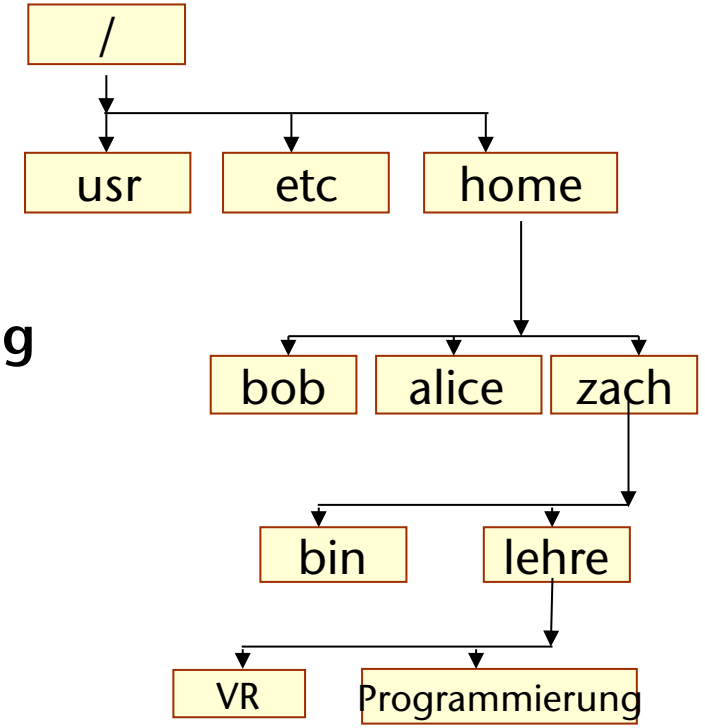
  →Eindeutigkeit von Pfadnamen garantiert

- Beispiele:


/
bin
dev
usr
etc
home
/bin/ls
ls
cp
/bin/cp
bin
man
local
bob
alice
zach
bin
lehre
/home/zach/lehre/Programmierung
VR
Programmierung

Absolute / relative Pfade

- Absolute Pfadnamen: starten mit /
- Relative Pfadnamen:
  - starten von einem anderen Dir aus
  - Sind also *relativ* zu diesem Dir
- Beispiele: der absolute Pfad **/home/zach/lehre/Programmierung** von ...
  - **home** aus = **zach/lehre/Programmierung**
  - **zach** aus = **lehre/Programmierung**
  - **lehre** aus = **Programmierung**



Spezielle Verzeichnisse

- '**.**' Bezeichnet das aktuelle Verzeichnis
  - Bsp.: **/bin/ls** = **/bin/./ls** = **/bin/././ls** ...
- '..' Bezeichnet das Vater-Verzeichnis (*parent diretory*)
  - Bsp.: **/usr/bin/w** = **/home/../usr/bin/w** = **/usr/man/../bin/w** ...
- Wird besonders wichtig im Zusammenhang mit dem CWD (*current working directory*)

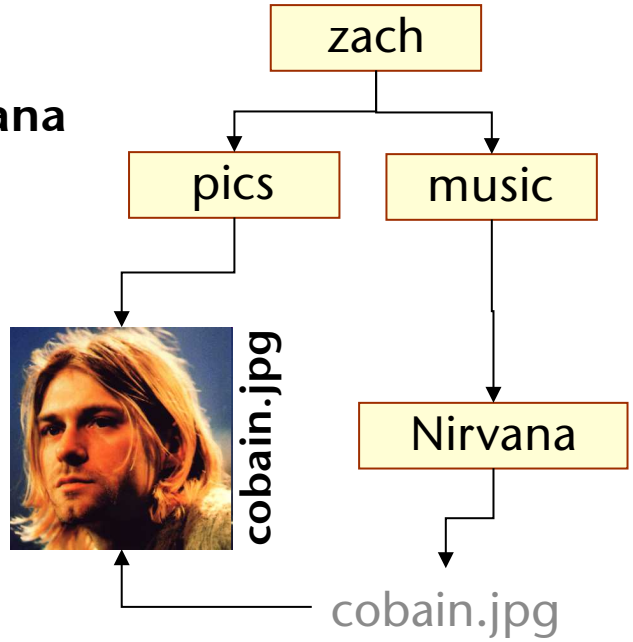
Kommandos: File- und Verzeichnis-Manipulation

Kommando	Funktion
rm file	File löschen
ls [dir]	Verzeichnis / File anzeigen
ls –l [dir]	Mehr Infos zum Verzeichnis / File anzeigen
ls –a [dir]	Dot-Files (.*) anzeigen
cp file1 ... dir	Files kopieren
cp file1 file2	Kopie von File1 erzeugen und File2 nennen
mv file1 ... dir	Files verschieben
mv file1 file2	File umbenennen
cat file1 file2 ... > file	Files aneinanderhängen (konkatenieren)
mkdir dir	Neues Verzeichnis erzeugen
rmdir dir	Verzeichnis löschen (muß leer sein)
touch file	Leeren File erzeugen

- Achtung: ES GIBT KEIN RECYCLE-BIN!!! ...

Symbolische Links

- Problem: File "gehört" genau einem Verzeichnis
  - Beispiel: File **/home/zach/pics/cobain.jpg** soll auch im Dir. **/home/zach/music/Nirvana** sichtbar sein ...
- Lösung: *symbolic links* (*symlinks*)
  - Bsp.: **music/Nirvana/cobain.jpg** ist ein Symlink nach **../../pics/cobain.jpg**



Kommando	Funktion
ln –s file1 file2	Erzeugt symbolischen Link von File2 nach File1 (Eselsbrücke: **ln –s** statt **cp**)
rm symlink	Löscht den Symbolic Link, nicht den File worauf dieser zeigt

Das *Current Working Directory*

- Die Shell merkt sich ein *Current Working Directory (CWD, PWD)*
  - Bei mehreren offenen Terminal-Fenstern (= Shells) merkt sich jede Shell ihr eigenes CWD
- Alle relativen Pfade werden von der Shell relativ zu diesem CWD interpretiert

- Für die Fortgeschrittenen:
  - Eigentlich hat jeder Prozeß sein eigenes CWD
    - (Auch die Shell ist ein ganz normaler Prozeß)
  - Die Interpretation eines relativen Pfades relativ zum CWD geschieht durch den Unix Kernel

Kommandos: Moving Around

Utility	Funktion
cd *dir*	Ins Verzeichnis **dir** wechseln (rel. oder abs. Pfad)
cd -	Ins vorige Verzeichnis zurück wechseln
cd	Ins Home wechseln
pwd	Aktuelles Verzeichnis (current working directory) anzeigen

Home Sweet Home

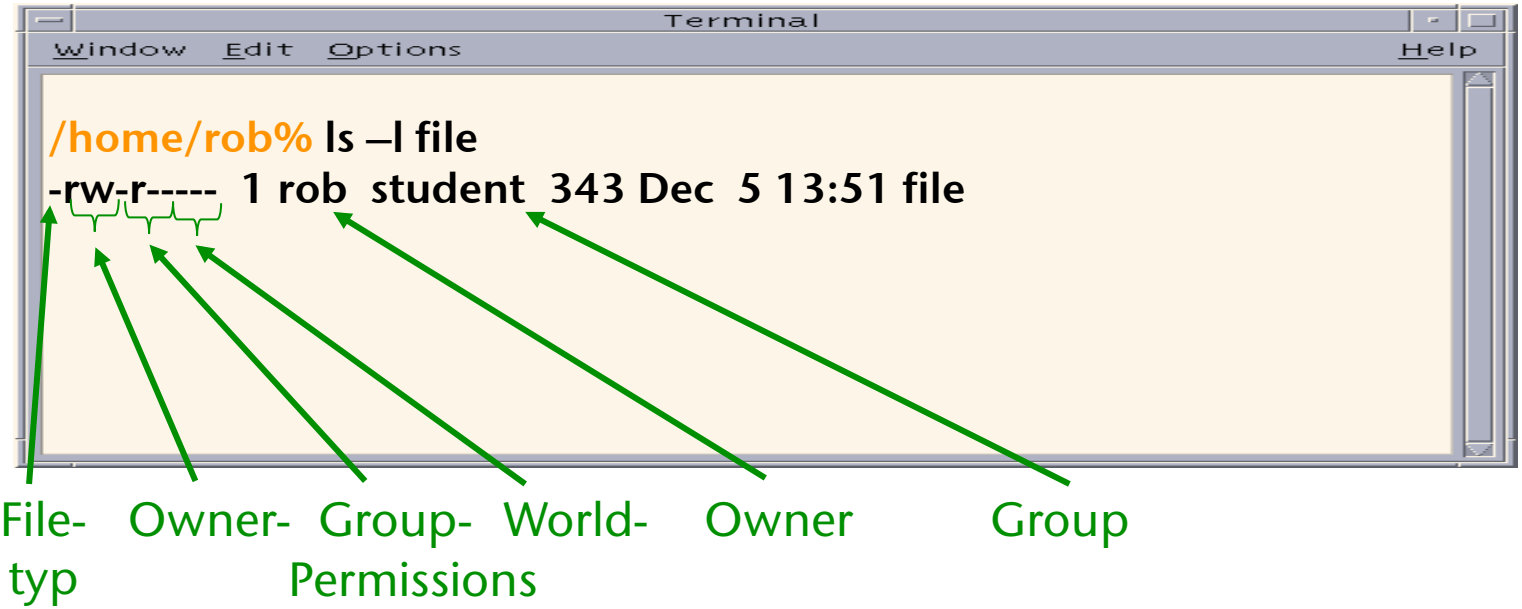
- Jeder User hat ein *Home*
  - Z.B. **/home/zach**
  - Enthält normalerweise alle Daten des Users
  - Alle Konfigurationsfiles aller Programme ("Dot-Files", z.B. **.login**) (riesiger Vorteil gegenüber Registry!)
- Beim Einloggen "startet man im Home" (d.h., CWD = ~)
- Normalerweise auf einem Fileserver
- Ist auf jeder Maschine gleich zugreifbar
- Schreibweise: ~

Users & Groups

- Daten eines Users:
  - Username (login, oft gleich wie email)
  - UID = ID des Usernames (**id** Kommando)
  - GID = group ID (evtl. mehrere)
  - Ein Home
  - Wird i.A. LAN-weit verwaltet
- Gruppen:
  - Jeder User gehört zu mindestens einer Gruppe
  - LAN-weit oder lokal

File Permissions

- 3 Personengruppen: Owner (=User), Group, World (Other)
- File gehört genau 1 User
- File ist assoziiert zu genau 1 Group
- Für jede der 3 Gruppen einen Satz File-Permissions:
  read, write, execute



- Filetyp-Flag:
  - Kein Permissionflag!
  - Zeigt Filetyp an:
    - **-** = normaler File
    - **d** = Directory
    - **l** = Symlink
    - ... einige seltenere Spezial-Flags
- Bedeutung der Permissions

Perm.	File	Directory
r (read)	Read a file	List files in ...
w (write)	Write a file	Create / move / remove a file in ...
x (execute)	Execute a file (shell script or binary)	Access a file in ...

  - Weitere, sehr praktische Flags (set-GID, set-UID, sticky, ...)

Permissions modifizieren

- Syntax von **chmod** ("change mode"):

chmod <*level***><***op***><***perm***>** *filename*

level = String aus: u, g, o, a (user, group, other, all)
op = ein Zeichen aus +, -, = (gets, loses, equals)
perm = String aus: r, w, x, ... (read, write, execute, ...)

- Beispiele:

```
% chmod u+x foobar
% chmod u+rwx,go-w foobar
% chmod g=u temp/
% chmod u=rwx,g=rwx,o= shared/
```