



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 Ifl:
 - 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/...
```

G. Zachmann Werkzeuge der Informatik - WS 07/08 Einführung in Unix 5


Was ist ein Betriebssystem?

- Vermittler, Manager, Ressourcen-Verwalter, ...




Hardware


terminal
keyboard
cpu
memory
printer
modem
etc.



Operating System



Controls both hardware and software



Software

compiler
editor
word processor
database
browser
HTML editor
image editor
etc.

G. Zachmann Werkzeuge der Informatik - WS 07/08 Einführung in Unix 6



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
- ...



Die Erfinder

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



Ken Thompson and Dennis Ritchie
Your new heroes



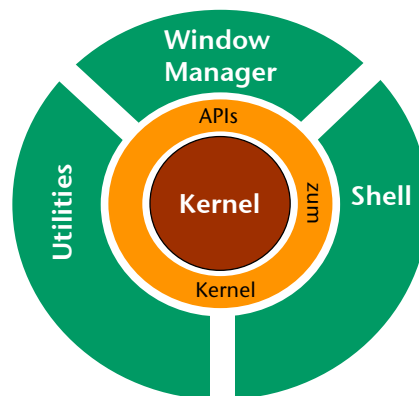
Was ist UNIX?

- 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



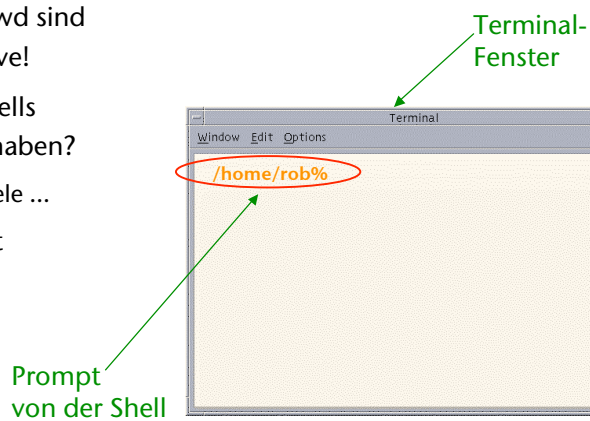
- Große Unsitte
 - Wegen Terminologie
- Also: **setenv LANG en** und **setenv KDE_LANG de**
 - In der bash: **export LANG=en** und **export KDE_LANG=de**
- Bzw. unter Linux: auf dem Login-Screen Englisch einstellen
- Oder: KDE Control Center → Regional & Accessibility → Country/Region & Language



Erstes Einloggen



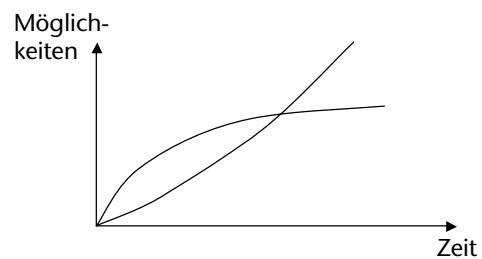
- Wie bekommt man eine Shell / (Terminal-)Fenster?
 - An der "Konsole" ("console")
 - Remote (ssh, rlogin, telnet)
- Login/passwd sind case-sensitive!
- Wieviele Shells kann man haben?
 - Beliebig viele ...
- Das Prompt





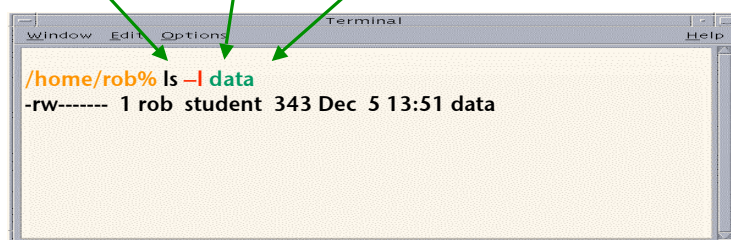
Das User-Interface

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



Aufbau einer Kommandozeile

Kommando Optionen Parameter



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



Remote Login



- Ist oft sehr praktisch! Z.B., wenn ...
 - ... auf dem aktuellen Rechner bestimmte Software nicht installiert ist
 - ... man einen anderen Rechner administrieren muß
- Befehl lautet **ssh**
- Klappt sogar mit GUIs
- Beispiel:

```
ssh as.rz.tu-clausthal.de -l gza
```

und, falls GUIs remote verwendet werden sollen,

```
ssh as.rz.tu-clausthal.de -l gza -X
```

- Zum Hin- und Her-Kopieren:

```
scp file user@remote.host:/path/to/file
```



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**)

G. Zachmann Werkzeuge der Informatik - WS 07/08 Einführung in Unix 19

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
 - ...

The diagram illustrates the UNIX architecture. At the center is a brown circle labeled "Kernel". Surrounding it are four green rectangular boxes: "Processes (time sharing, protected address space)" at the top, "Interprocess comm. (signals, pipes sockets, ...)" on the left, "Virtual memory (swapping, paging, mapping)" on the right, and "The filesystem (files, directories, devices, pipes, namespace, ...)" at the bottom.

G. Zachmann Werkzeuge der Informatik - WS 07/08 Einführung in Unix 20



Das Filesystem

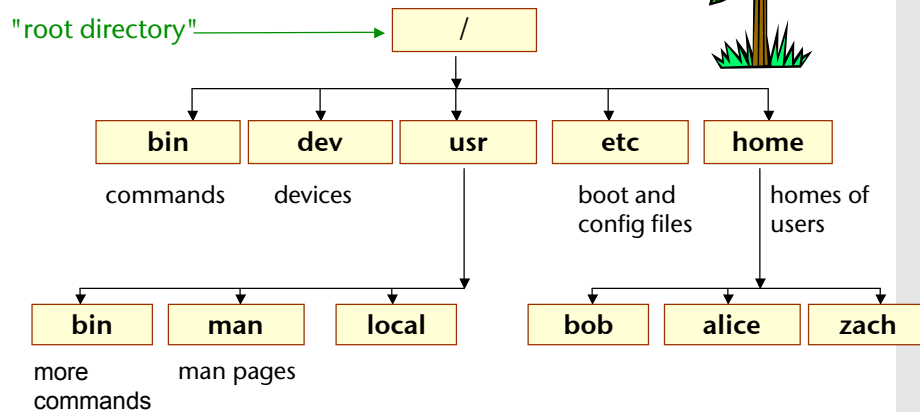


- Directories ("Folders") und Files
- File enthält sequentielle Folge von Zeichen (Bytes)
- Interpretation ist Sache des benutzenden Programms:
 - Text, Zahlen, Programm, Speicherauszug, ...
- Jeder File hat einen Namen:
 - Case-sensitive! (UNIX allg.)
 - Länge typ. bis zu 1024
 - Können beliebige Zeichen 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

File Tree



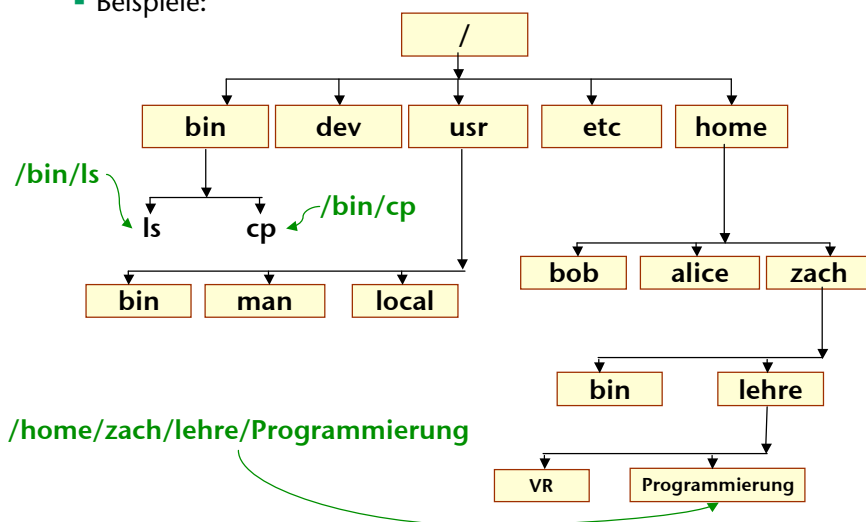


Eindeutigkeit

- 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:



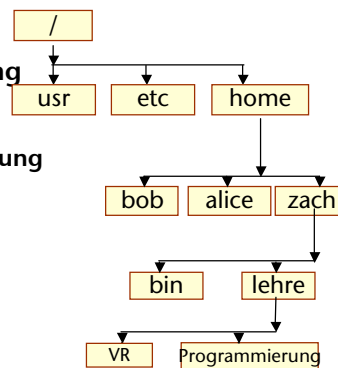


Absolute / relative Pfade

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

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

Kleine Warnung zu rm

Task: Shoot Yourself in The Foot

The proliferation of modern programming languages (all of which seem to have stolen countless features from one another) sometimes makes it difficult to remember what language you're currently using. This handy reference is offered as a public service to help programmers who find themselves in such a dilemma.

```
% ls
foot.c foot.h foot.o toe.c toe.o
% rm * .o
rm: .o no such file or directory
% ls
%
```



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 |
|--------------------------------|---|
| <code>ln -s file1 file2</code> | Erzeugt symbolischen Link von File2 nach File1 (Eselsbrücke: <code>ln -s</code> statt <code>cp</code>) |
| <code>rm symlink</code> | Löscht den Symbolic Link, nicht den File worauf dieser zeigt |

G. Zachmann Werkzeuge der Informatik - WS 07/08 Einführung in Unix 29

Andere Platten

- Der Verzeichnisbaum enthält (i.A.) mehrere Platten!
- Einige davon sind auf anderen Rechnern (NFS)

G. Zachmann Werkzeuge der Informatik - WS 07/08 Einführung in Unix 30



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 |
|---------------------|--|
| <code>cd dir</code> | Ins Verzeichnis dir wechseln (rel. oder abs. Pfad) |
| <code>cd -</code> | Ins vorige Verzeichnis zurück wechseln |
| <code>cd</code> | Ins Home wechseln |
| <code>pwd</code> | 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
 - LAN-weit verwaltet oder lokal
- 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

```
Terminal
Window Edit Options Help
/home/rob% ls -l file
-w-r----- 1 rob student 343 Dec 5 13:51 file
```

File-
typ

Owner-
Permissions

Group-
Permissions

World-
Permissions

Owner

Group



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



Exkurs: ACLs



- Oft feinere / flexiblere Regelung der Zugriffsrechte gewünscht
- ACLs = access control lists
- Features:
 - Individuelle Permissions pro User möglich
 - Selbst-definierte Gruppen
 - Permissions pro selbst-definierter Gruppe
 - ...
- Für rel. kleine Arbeitsgruppen ist das "normale" Unix-Permissions-Modell völlig ausreichend



Weitere File-Attribute

- Zeiten:
 - Modification (write): `ls -l`
 - Creation: `ls -lc`
 - Access (read): `ls -lu`

```
Terminal
Window Edit Options Help
/home/rob% ls -l file
-rw-r----- 1 rob student 343 Dec 5 13:51 file
```

- Größe, Links, ...

mod time



Prozesse

- Programm, das gerade läuft, schläft, oder hängt
- Jeder Prozeß führt ein *Environment* mit sich:
 - Prozeß-ID (PID), User-ID (UID), Group-ID (GID), u.a. IDs
 - current working directory (CWD, manchmal auch PWD)
 - Environment-Variablen (Paare von Strings)
 - ...
- Relative Pfade werden bzgl. des CWD's des Prozesses interpretiert



Spawning processes

- Ein Prozeß wurde *immer* von einem anderen erzeugt
- Heißt Vater-Prozeß (*parent process*)
- Vorgang heißt engl. *to spawn*
- *Child process* erbt das komplette Environment (außer seinen IDs u.ä.)

| Process ID | Process Name | User | CPU | Private Memory | Virtual Memory |
|------------|------------------|---------|------|----------------|----------------|
| 0 | kernel_task | root | 0.00 | 0 | 0 |
| 1 | launchd | root | 0.00 | 3 | 516,00 KB |
| 11 | WindowServer | root | 0.00 | 1 | 224,00 KB |
| 81 | WindowServer | awshome | 0.00 | 3 | 51,76 KB |
| 118 | Activity Monitor | asch1 | 0.00 | 4 | 224,00 KB |
| 188 | perl | root | 0.00 | 1 | 1,07 KB |
| 214 | perlFront | asch1 | 0.00 | 1 | 94,03 KB |
| 218 | perl | asch1 | 0.00 | 3 | 1,07 KB |
| 241 | perl | asch1 | 0.00 | 8 | 10,17 KB |
| 242 | perl | asch1 | 0.00 | 3 | 1,07 KB |
| 243 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 244 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 245 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 246 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 247 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 248 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 249 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 250 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 251 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 252 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 253 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 254 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 255 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 256 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 257 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 258 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 259 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 260 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 261 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 262 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 263 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 264 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 265 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 266 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 267 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 268 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 269 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 270 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 271 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 272 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 273 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 274 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 275 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 276 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 277 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 278 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 279 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 280 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 281 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 282 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 283 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 284 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 285 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 286 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 287 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 288 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 289 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 290 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 291 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 292 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 293 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 294 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 295 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 296 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 297 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 298 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 299 | perl | asch1 | 0.00 | 3 | 2,24 KB |
| 300 | perl | asch1 | 0.00 | 3 | 2,24 KB |

```

ps
  PID TTY          PPID  %CPU   MEMRES  VSZ
  ---  ---          ---  ---    ---     ---
  300  ttys000    299   0.0  1024K   12K
  301  ttys000    299   0.0  1024K   12K
  302  ttys000    299   0.0  1024K   12K
  303  ttys000    299   0.0  1024K   12K
  304  ttys000    299   0.0  1024K   12K
  305  ttys000    299   0.0  1024K   12K
  306  ttys000    299   0.0  1024K   12K
  307  ttys000    299   0.0  1024K   12K
  308  ttys000    299   0.0  1024K   12K
  309  ttys000    299   0.0  1024K   12K
  310  ttys000    299   0.0  1024K   12K
  311  ttys000    299   0.0  1024K   12K
  312  ttys000    299   0.0  1024K   12K
  313  ttys000    299   0.0  1024K   12K
  314  ttys000    299   0.0  1024K   12K
  315  ttys000    299   0.0  1024K   12K
  316  ttys000    299   0.0  1024K   12K
  317  ttys000    299   0.0  1024K   12K
  318  ttys000    299   0.0  1024K   12K
  319  ttys000    299   0.0  1024K   12K
  320  ttys000    299   0.0  1024K   12K
  321  ttys000    299   0.0  1024K   12K
  322  ttys000    299   0.0  1024K   12K
  323  ttys000    299   0.0  1024K   12K
  324  ttys000    299   0.0  1024K   12K
  325  ttys000    299   0.0  1024K   12K
  326  ttys000    299   0.0  1024K   12K
  327  ttys000    299   0.0  1024K   12K
  328  ttys000    299   0.0  1024K   12K
  329  ttys000    299   0.0  1024K   12K
  330  ttys000    299   0.0  1024K   12K
  331  ttys000    299   0.0  1024K   12K
  332  ttys000    299   0.0  1024K   12K
  333  ttys000    299   0.0  1024K   12K
  334  ttys000    299   0.0  1024K   12K
  335  ttys000    299   0.0  1024K   12K
  336  ttys000    299   0.0  1024K   12K
  337  ttys000    299   0.0  1024K   12K
  338  ttys000    299   0.0  1024K   12K
  339  ttys000    299   0.0  1024K   12K
  340  ttys000    299   0.0  1024K   12K
  341  ttys000    299   0.0  1024K   12K
  342  ttys000    299   0.0  1024K   12K
  343  ttys000    299   0.0  1024K   12K
  344  ttys000    299   0.0  1024K   12K
  345  ttys000    299   0.0  1024K   12K
  346  ttys000    299   0.0  1024K   12K
  347  ttys000    299   0.0  1024K   12K
  348  ttys000    299   0.0  1024K   12K
  349  ttys000    299   0.0  1024K   12K
  350  ttys000    299   0.0  1024K   12K
  351  ttys000    299   0.0  1024K   12K
  352  ttys000    299   0.0  1024K   12K
  353  ttys000    299   0.0  1024K   12K
  354  ttys000    299   0.0  1024K   12K
  355  ttys000    299   0.0  1024K   12K
  356  ttys000    299   0.0  1024K   12K
  357  ttys000    299   0.0  1024K   12K
  358  ttys000    299   0.0  1024K   12K
  359  ttys000    299   0.0  1024K   12K
  360  ttys000    299   0.0  1024K   12K
  361  ttys000    299   0.0  1024K   12K
  362  ttys000    299   0.0  1024K   12K
  363  ttys000    299   0.0  1024K   12K
  364  ttys000    299   0.0  1024K   12K
  365  ttys000    299   0.0  1024K   12K
  366  ttys000    299   0.0  1024K   12K
  367  ttys000    299   0.0  1024K   12K
  368  ttys000    299   0.0  1024K   12K
  369  ttys000    299   0.0  1024K   12K
  370  ttys000    299   0.0  1024K   12K
  371  ttys000    299   0.0  1024K   12K
  372  ttys000    299   0.0  1024K   12K
  373  ttys000    299   0.0  1024K   12K
  374  ttys000    299   0.0  1024K   12K
  375  ttys000    299   0.0  1024K   12K
  376  ttys000    299   0.0  1024K   12K
  377  ttys000    299   0.0  1024K   12K
  378  ttys000    299   0.0  1024K   12K
  379  ttys000    299   0.0  1024K   12K
  380  ttys000    299   0.0  1024K   12K
  381  ttys000    299   0.0  1024K   12K
  382  ttys000    299   0.0  1024K   12K
  383  ttys000    299   0.0  1024K   12K
  384  ttys000    299   0.0  1024K   12K
  385  ttys000    299   0.0  1024K   12K
  386  ttys000    299   0.0  1024K   12K
  387  ttys000    299   0.0  1024K   12K
  388  ttys000    299   0.0  1024K   12K
  389  ttys000    299   0.0  1024K   12K
  390  ttys000    299   0.0  1024K   12K
  391  ttys000    299   0.0  1024K   12K
  392  ttys000    299   0.0  1024K   12K
  393  ttys000    299   0.0  1024K   12K
  394  ttys000    299   0.0  1024K   12K
  395  ttys000    299   0.0  1024K   12K
  396  ttys000    299   0.0  1024K   12K
  397  ttys000    299   0.0  1024K   12K
  398  ttys000    299   0.0  1024K   12K
  399  ttys000    299   0.0  1024K   12K
  400  ttys000    299   0.0  1024K   12K
  
```

top in der Shell



Prozesse aus Sicht der Shell

- 3 Zustände eines Prozesses (aus Sicht der Shell)
 - Foreground: Default
 - Ausgabe (stdout) des Prozesses erscheint im Terminal-Fenster
 - Eingabe (stdin) des Prozesses kommt vom Keyboard
 - Background:
 - Ausgabe erscheint im Fenster
 - Eingabe nicht erlaubt
 - Gestoppt:
 - Prozeß schläft



Kommandos zur Prozeßkontrolle



| Befehl | Funktion |
|--------------------------------|---|
| <code>ps</code> | Prozesse anzeigen |
| <code>ps -edfjw</code> | Alle Prozesse anzeigen |
| <code>ps -auxw</code> | dito für einige andere Unix-Varianten |
| <code>kill pid</code> | Prozeß mit PID <code>pid</code> abbrechen (wie Ctrl-C) |
| <code>kill -9 pid</code> | ... wenn der Prozeß trotzdem nicht aufhören will ☹ |
| <code>command ... &</code> | Prozeß im Hintergrund starten |
| <code>jobs</code> | Prozesse im Hintergrund anzeigen |
| <code>Ctrl-C</code> | Foreground-Prozeß abbrechen (interrupt) |
| <code>Ctrl-Z</code> | Foreground-Prozeß anhalten (stoppen) |
| <code>fg</code> | Zuletzt angehaltenen Prozeß im Foreground weiterlaufen lassen |
| <code>bg</code> | Angehaltenen Prozeß im Background weiterlaufen lassen |
| <code>Ctrl-S</code> | Ausgabe des Foreground-Prozesses anhalten (Pr. läuft weiter!) |
| <code>Ctrl-Q</code> | Ausgabe weiterlaufen lassen |
| <code>top</code> | tabellarische Ansicht aller Prozesse und deren CPU-Verbrauch |