

# Linux-Kurs – Teil 1

FSI Informatik

FAU Erlangen-Nürnberg

17. Oktober 2023

### Mittwoch 19.04.2023

<b>Zeit</b>	<b>Raum</b>	<b>Inhalt</b>
10:00–12:00	H16	Vorlesung (Teil 1)
12:00–14:00	CIP2	Übung (Teil 1)

### Donnerstag 20.04.2023

<b>Zeit</b>	<b>Raum</b>	<b>Inhalt</b>
10:00–12:00	H16	Vorlesung (Teil 2)
12:00–14:00	CIP2	Übung (Teil 2)

- ▶ Eigentlich nur ein Betriebssystemkern
- ▶ Üblicherweise kombiniert mit Standardwerkzeugen aus dem GNU-Projekt
  - Deshalb auch *GNU/Linux* genannt
  - Geht aber auch anders, z. B. Android
- ▶ *Unix-artiges* Betriebssystem
  - Ursprüngliches Unix heutzutage nicht mehr relevant
  - Heutige „Verwandte“: BSD, Mac OS X

# Allgemeines

## Linux-Distributionen

- ▶ Meistens meint man mit *Linux* eine Zusammenstellung von:
  - Betriebssystem
  - (Arbeits-)Programmen
  - Benutzeroberfläche
- ▶ Diese *Linux-Distributionen* haben eigene Namen und Versionsnummern, z. B.:



- **debian** (hier im CIP installiert)

- **ubuntu**<sup>®</sup>



-  **Gentoo**

- ...



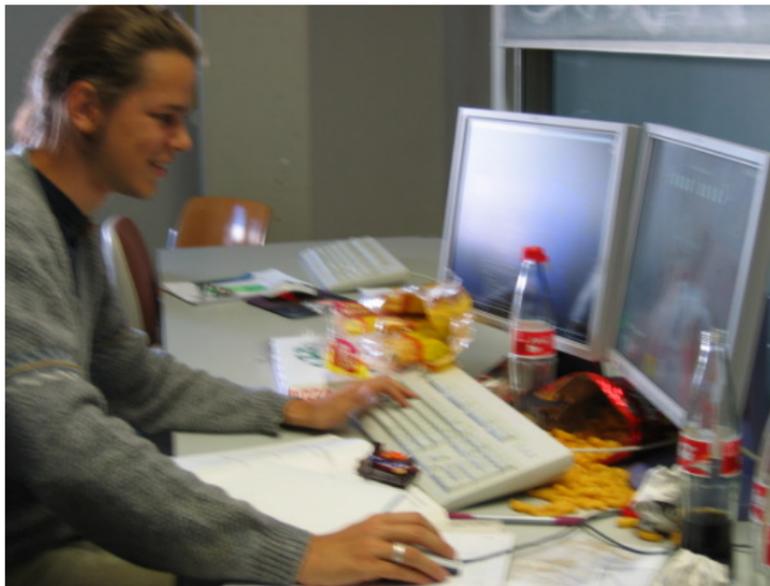
### CIP-Pools im Blauen Hochhaus:

- ▶ Linux-Arbeitsrechner
- ▶ Drucker
- ▶ Farbdrucker-Scanner-  
Multifunktions-Monster  
(im CIP 2)



# Allgemeines

Warnung!



Essen und Trinken verboten!  
(Loginentzug droht)

# Allgemeines

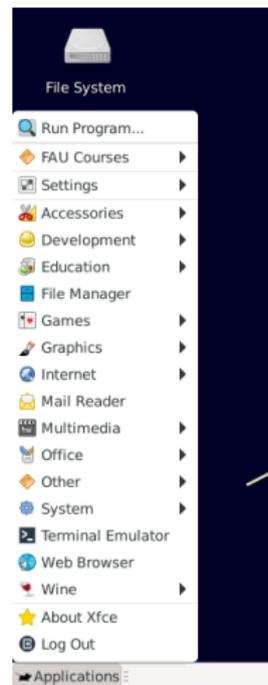
## Window-Manager – XFCE

### Window-Manager

Bestimmt Aussehen und Verhalten der graphischen Oberfläche



- ▶ Gut geeignet für den Einstieg
- ▶ Thunar (Dateimanager)
- ▶ Webbrowser
- ▶ *System*-Menü zur Konfiguration
- ▶ Übersichtliche schlanke Oberfläche



- ▶ Intuitive Bedienung („ähnlich wie unter Windows“)
- ▶ In der Standardeinstellung komplett auf englisch – aber das solltet ihr alle können. . .
- ▶ Wir trauen euch zu, dass ihr selbstständig zurecht kommt 😊
- ▶ Daher: in diesem Kurs Konzentration auf Befehlszeile & Co.

### Gibt's trotzdem Probleme?

Universeller Lösungsalgorithmus: <https://xkcd.com/627/>

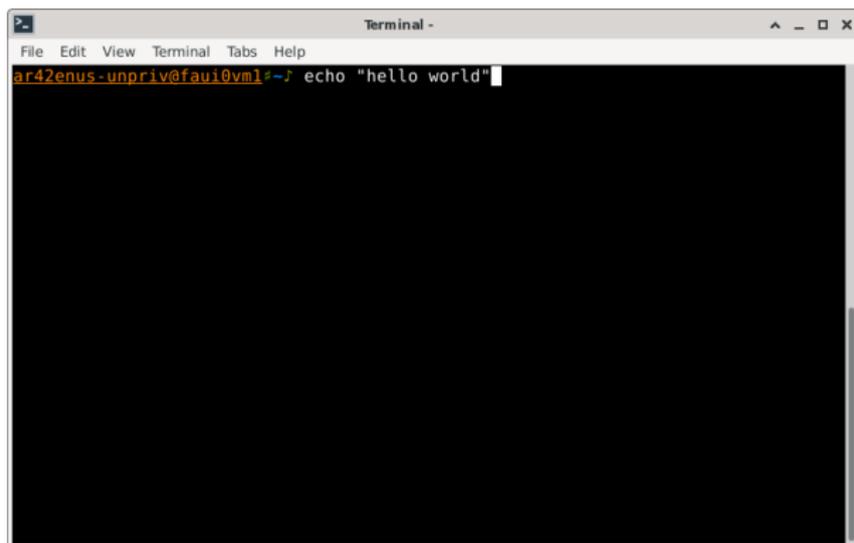
# Allgemeines

## Bildschirm sperren und abmelden



# Allgemeines

## Befehlszeile – Warum?

A screenshot of a terminal window titled "Terminal -". The window has a menu bar with "File", "Edit", "View", "Terminal", "Tabs", and "Help". The terminal content shows the prompt "ar42enus-unpriv@fau10vm1:~#" followed by the command "echo \"hello world\"". The cursor is positioned at the end of the command. The rest of the terminal area is black and empty.

```
Terminal -
File Edit View Terminal Tabs Help
ar42enus-unpriv@fau10vm1:~# echo "hello world"
```

Getippte Befehle anstelle graphischer Anwendungen.

**Warum?! Ist das nicht ein riesiger Rückschritt?**

# Allgemeines

## Verkleinern eines Bildes

Beispiel: Verkleinern eines Bildes

1. Graphikprogramm aus dem Startmenü ausführen.
2. *Datei* → *Öffnen* klicken.
3. Den richtigen Ordner suchen.
4. Die Bilddatei auswählen.
5. Im *Bild*-Menü auf den Befehl *Skalieren* klicken.
6. Die neue Größe eingeben.
7. *Datei* → *Speichern unter* klicken.
8. Den neuen Dateinamen eingeben.



Und auf der Befehlszeile?

Wenn man erst einmal weiß wie, genügt ein Befehl<sup>1</sup>:

```
$ convert -resize 300 gnu.png gnu-klein.png
```

Und das kann man auch **mit einem einzigen Befehl** für 100 Dateien durchführen!

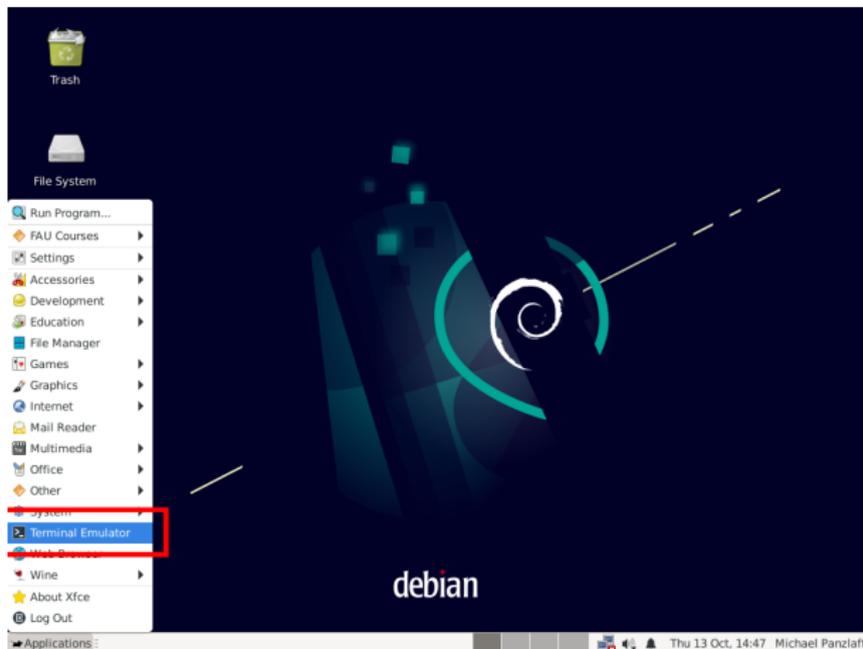
- ▶ Zwar höhere Einarbeitungszeit. . .
- ▶ . . . aber auf Dauer deutlich schneller!
- ▶ und einfach auf anderen Rechnern im Netzwerk benutzbar
- ▶ und automatisierbar!

<sup>1</sup>\$ ist das sogenannte *Prompt*-Symbol und muss nicht mit eingetippt werden.

# Terminal

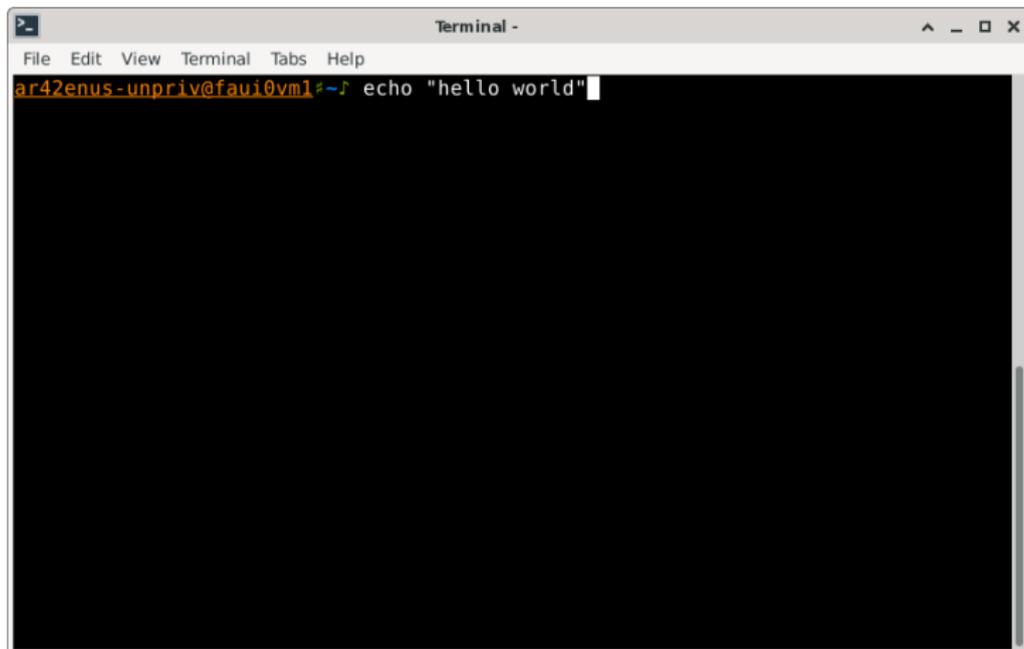
Und wo kann ich diese Befehle eingeben?

Das passende Programm von XFCE, der Standard-Desktop-Umgebung im CIP, heißt *Terminal*:



# Terminal

... und sieht so aus:



The image shows a terminal window titled "Terminal -". The window has a menu bar with "File", "Edit", "View", "Terminal", "Tabs", and "Help". The terminal prompt is "ar42enus-unpriv@fau10vm1#~". The command "echo \"hello world\"" is entered and the cursor is at the end of the line. The terminal background is black.

```
Terminal -
File Edit View Terminal Tabs Help
ar42enus-unpriv@fau10vm1#~$ echo "hello world"
```

### Shell

- ▶ Programm, welches eingetippte Befehle entgegennimmt
- ▶ **bash** ist die Standardshell im CIP

Im Terminal kann man jetzt Befehle eingeben:

```
$ echo
```

echo gibt den übergebenen Text unverändert wieder aus.

# Befehlsaufbau

## Befehle mit einem Parameter

Dazu brauchen wir Parameter:

### Muster

*<Befehl> <Parameter>*

```
$ echo foo  
foo
```

# Befehlsaufbau

## Mehrere Parameter

Also einmal mit zwei Wörtern:

```
$ echo foo bar  
foo bar
```

... und noch ein paar Leerzeichen mehr:

```
$ echo foo    bar  
foo bar
```

# Befehlsaufbau

## Quoting

### Problem:

```
$ echo foo      bar
foo bar
```

Mehrere Parameter werden durch Leerzeichen getrennt – wie viele Leerzeichen, spielt keine Rolle.

Durch *Quoting* kann man die Spezialbedeutung von Leerzeichen<sup>2</sup> aufheben – der Text, der in Anführungszeichen steht, wird als ein einziger langer Parameter interpretiert.

### Lösung:

```
$ echo 'foo      bar'
foo      bar
```

<sup>2</sup>und anderen Sonderzeichen

Je nach Befehl können auch verschiedene Optionen angegeben werden, um das Verhalten des Befehls zu verändern:

### Muster

```
<Befehl> <Optionen> <Parameter>
```

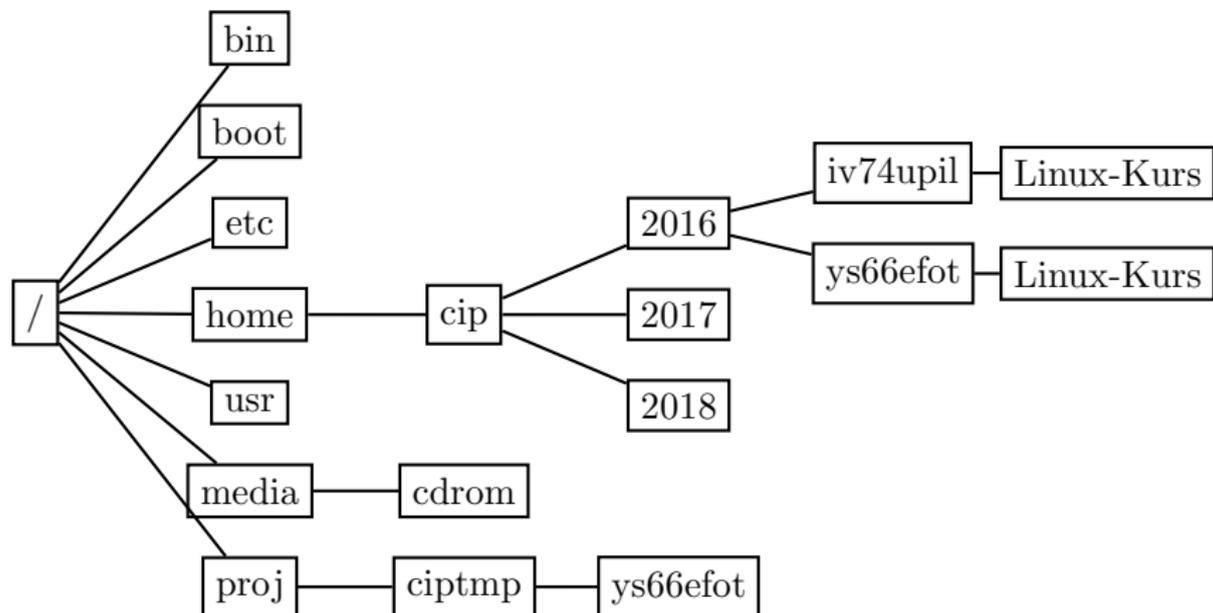
Bei `echo` bewirkt die Option `-n`, dass nach der Ausgabe keine neue Zeile angefangen wird.

```
$ echo -n foo  
foo $ _
```



# Herumklettern im Dateisystembaum

## Aufbau des Verzeichnisbaums



# Herumklettern im Dateisystembaum

## Unterschiede zu Windows

- ▶ Es gibt nur einen großen Dateisystembaum, nicht mehrere mit jeweils einem Laufwerksbuchstaben.
- ▶ Pfadtrenner: / (*Slash*) statt \ (*Backslash*).
- ▶ Zwischen Groß- und Kleinschreibung wird unterschieden!

```
C:\Users\klaus c:\users\KLaUs  
/home/klaus
```

# Herumklettern im Dateisystembaum

`cip-mountusb` – USB-Sticks einhängen

## USB im CIP

`cip-mountusb`

hängt den USB-Stick unter `/media/usb` ein

`cip-umountusb`

hängt den USB-Stick wieder aus

## Anmerkungen

- ▶ Vor dem Abziehen des Sticks unmounten → sonst Datenverlust!
- ▶ FAT32 und NTFS wird unterstützt

# Herumklettern im Dateisystembaum

Wo zum Teufel sind wir überhaupt?

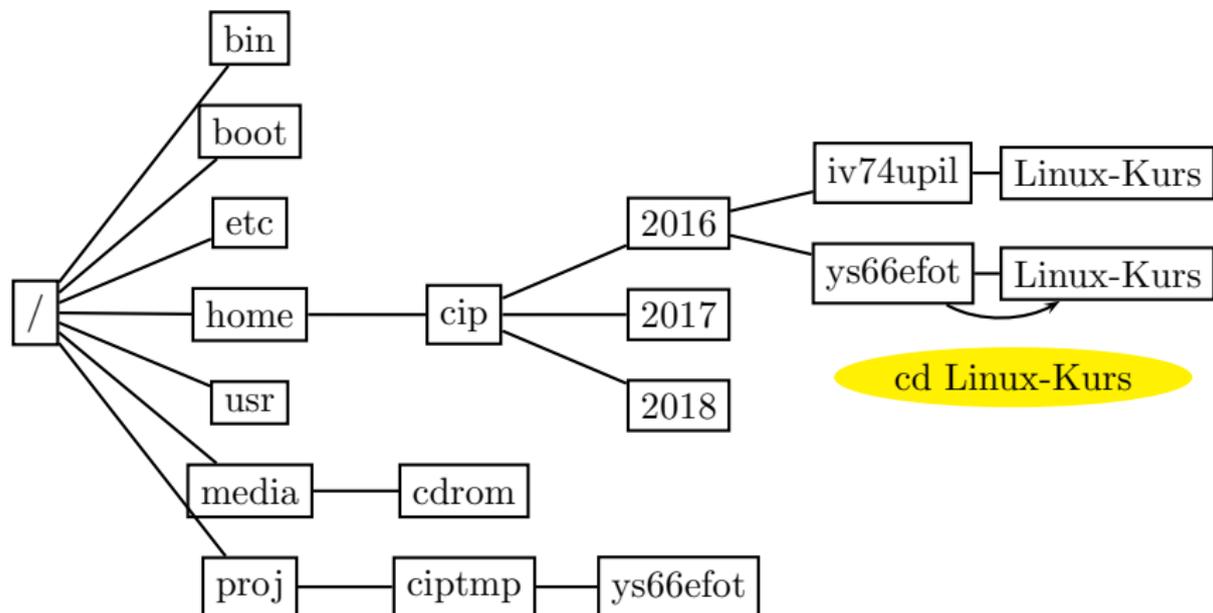
```
pwd
```

`pwd` (*print working directory*) gibt das aktuelle Verzeichnis aus.

```
$ pwd  
/home/cip/2016/ys66efot
```

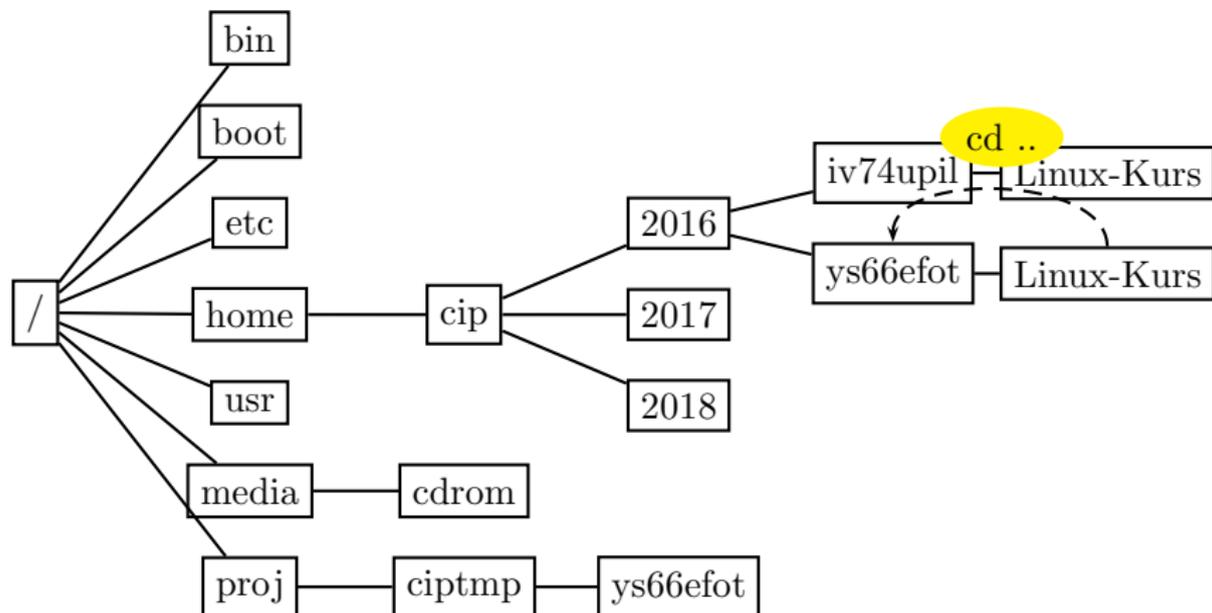
# Herumklettern im Dateisystembaum

## Verzeichniswechsel



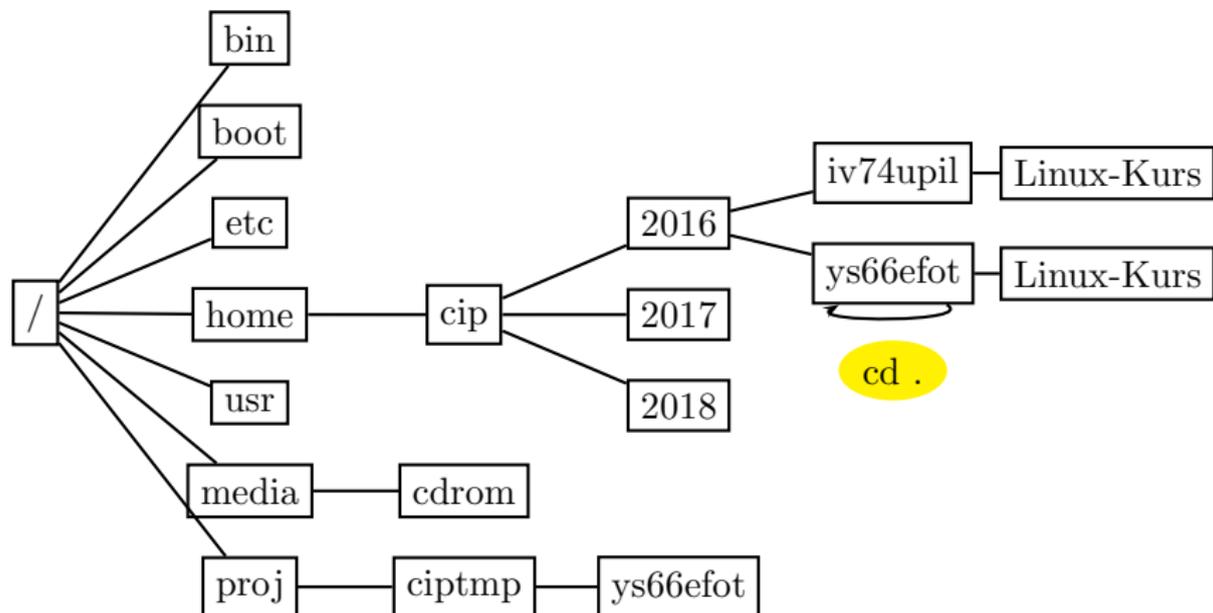
# Herumklettern im Dateisystembaum

Verzeichniswechsel ins übergeordnete Verzeichnis



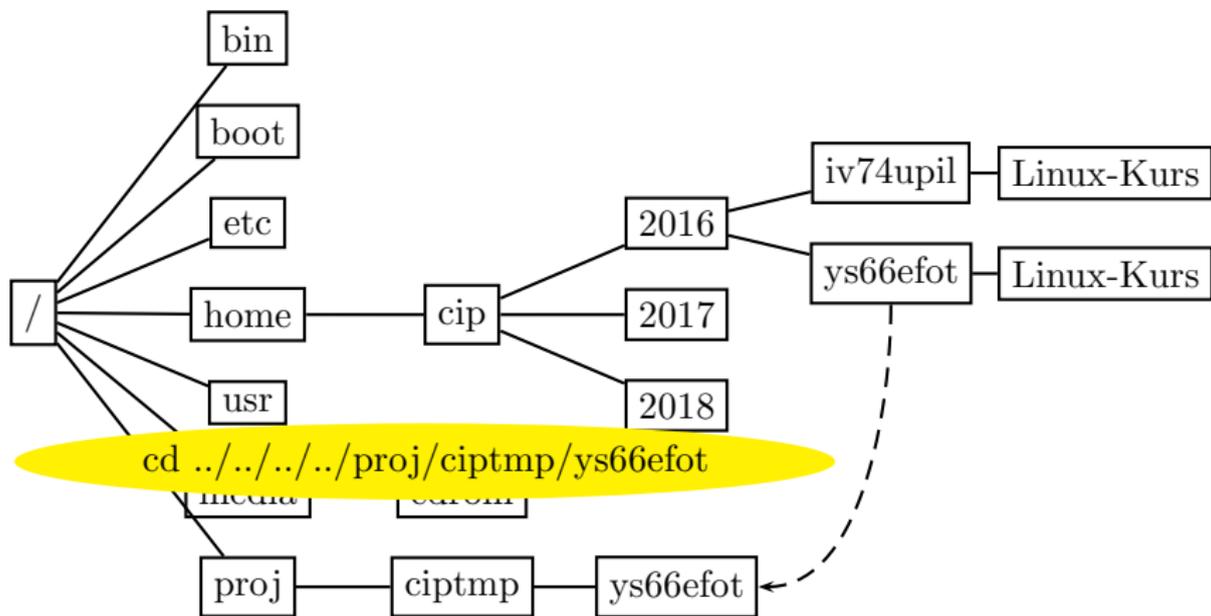
# Herumklettern im Dateisystembaum

„Verzeichniswechsel“



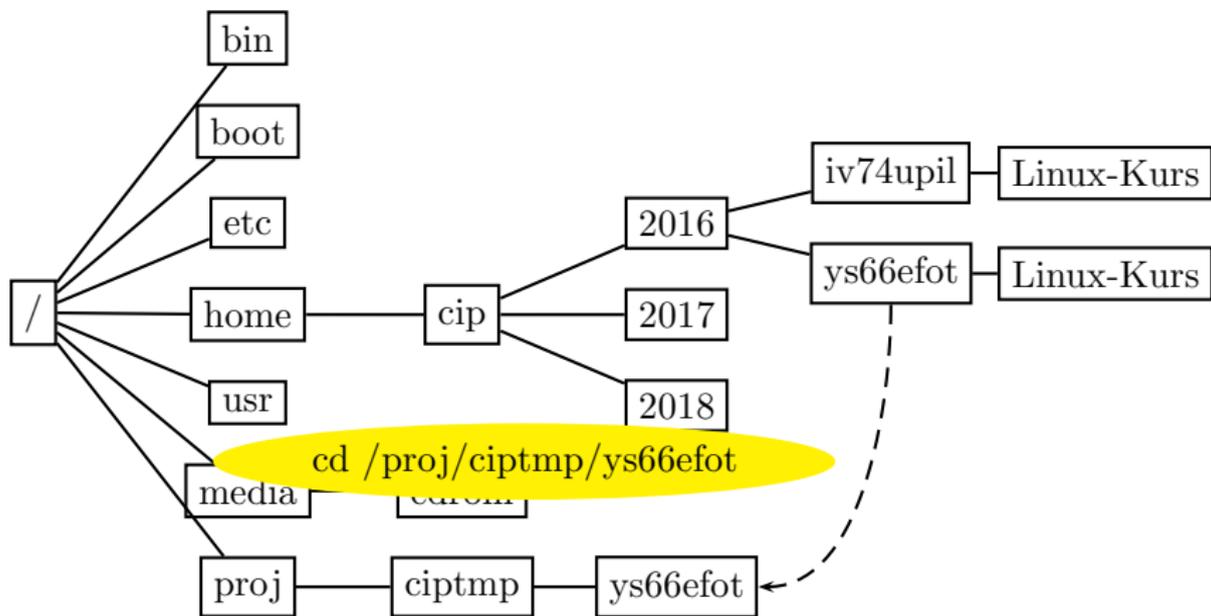
# Herumklettern im Dateisystembaum

Relativer Verzeichniswechsel (relativ zum aktuellen Verzeichnis)



# Herumklettern im Dateisystembaum

Absoluter Verzeichniswechsel (ausgehend vom Wurzelverzeichnis – vorangestellter /)



# Herumklettern im Dateisystembaum

## Verzeichniswechsel

### cd

Mit `cd` (= *change directory*) wechselt man zwischen Verzeichnissen.

### Beispiele

- |                          |   |  |
|--------------------------|---|--|
| <code>cd bin</code>      | – | wechselt in das Unterverzeichnis „bin“ im aktuellen Verzeichnis ( <i>relativer Pfadwechsel</i> )   |
| <code>cd /bin</code>     | – | geht in das Verzeichnis „bin“ unterhalb des Root-Verzeichnisses / ( <i>absoluter Pfadwechsel</i> ) |
| <code>cd ..</code>       | – | wechselt eine Verzeichnisebene nach oben   |
| <code>cd ../testy</code> | – | wechselt eine Verzeichnisebene nach oben <b>und</b> darin in das Verzeichnis „testy“               |
| <code>cd</code>          | – | geht in das <i>Home</i> -Verzeichnis   |
| <code>cd -</code>        | – | geht in das letzte besuchte Verzeichnis  |

# Herumklettern im Dateisystembaum

## Home und ciptmp

- ▶ Jeder Benutzer besitzt ein *Home*-Verzeichnis (`/home/cip/2022/<userlogin>`):
  - Es steht nur begrenzter Speicherplatz zur Verfügung
  - Dort liegen Konfigurationen und Nutzdaten
  - Der Inhalt wird täglich gesichert und ist zentral gespeichert, also auf allen Rechnern gleich
  - Kurzschreibweise fürs *Home*-Verzeichnis: `~` (*Tilde-Zeichen*)
  
- ▶ Mehr Speicherplatz (8 GB) ist im *ciptmp* verfügbar (`/proj/ciptmp/<userlogin>`):
  - Wird nicht gesichert und kann ohne Vorwarnung gelöscht werden!
  - Wird erst bei Betreten eingebunden (d. h. ein `ls` auf `/proj/` kann u. U. den Anschein erwecken, dass das Verzeichnis leer ist!)

Der Befehl `cip-quota` zeigt, wie viel Speicherplatz zur Verfügung steht.

# Herumklettern im Dateisystembaum

Speicherplatzverbrauch – per Konsole

## du

Mit `du` (= *disk usage*) kann man sich den Speicherplatz anzeigen lassen.

## Beispiele

- `du` – gibt den Speicherbedarf aller Dateien aus  
(rekursiv für jeden Ordner)
- `du -h` – `-h` = *human-readable*  
→ gibt die Größen besser lesbar aus
- `du -sh` – gibt den Speicherbedarf des aktuellen Ordners lesbar aus

# Herumklettern im Dateisystembaum

Speicherplatzverbrauch – interaktiv per Konsole mit `ncdu`

```
$ ncdu /etc
```

```
ncdu 1.15.1 ~ Use the arrow keys to navigate, press ? for help
--- /etc -----
 6.3 MiB [#####] /libreoffice
 1.4 MiB [##      ] /X11
 1.0 MiB [#       ] /scite
 1.0 MiB [#       ] /xdg
 800.0 KiB [#      ] /ssh
 760.0 KiB [#      ] /ssl
 712.0 KiB [#      ] /mono
 588.0 KiB [       ] /direct
 496.0 KiB [       ] avrdude.conf
 496.0 KiB [       ] /java-17-openjdk
 496.0 KiB [       ] /java-11-openjdk
 496.0 KiB [       ] /asciidoc
 484.0 KiB [       ] /timidity
 472.0 KiB [       ] ld.so.cache
 380.0 KiB [       ] /xpra
 340.0 KiB [       ] /sane.d
 312.0 KiB [       ] /joe
 288.0 KiB [       ] /init.d
 280.0 KiB [       ] /ImageMagick-6
 260.0 KiB [       ] /nagios-plugins
 252.0 KiB [       ] /systemd
 216.0 KiB [       ] /alternatives
Total disk usage: 25.8 MiB Apparent size: 20.0 MiB Items: 4320
```

# Inhalte aufzeigen

## Verzeichnisinhalt

ls

ls listet den Inhalt eines Verzeichnisses auf.

## Beispiele

- ls – listet Inhalt des aktuellen Verzeichnisses auf
- ls verzeichnis – listet Inhalt des angegebenen Verzeichnisses auf
- ls -d verzeichnis – gibt Informationen zum angegebenen Verzeichnis aus (nicht aber den Inhalt)
- ls -l – ausführliche Verzeichnisauflistung (Dateigrößen, Rechte, Zeitstempel etc.)
- ls -a – listet auch versteckte Dateien (Dateien, die mit einem Punkt beginnen) auf

# Inhalte aufzeigen

## Beispiele

### Normales `ls` vs. `ls -a`

```
$ ls
a.txt mein_bild.jpg

$ ls -a
.  ..  .bash_history  a.txt  mein_bild.jpg
```

- ▶ `ls -a` zeigt wirklich alle Einträge des Verzeichnisses an!
- ▶ Einträge, die mit einem `.` beginnen, werden normalerweise als *versteckt* interpretiert und nicht angezeigt, z. B.:
  - `.` ist immer das aktuelle Verzeichnis
  - `..` ist immer das übergeordnete Verzeichnis
  - `.bash_history` enthält z. B. Befehle, die früher eingegeben wurden

# Fahrt aufnehmen

## Tab-Vervollständigung

Mit einem Druck auf <TAB> wird u. a. Folgendes ergänzt:

- ▶ Namen von Befehlen
- ▶ Datei- und Verzeichnisnamen

```
$ ls
Desktop  folien_linuxkurs_tag1.pdf

$ file f<TAB>
$ file folien_linuxkurs_tag1.pdf
folien_linuxkurs_tag1.pdf: PDF document, version 1.4
```

# Fahrt aufnehmen

## Tab-Vervollständigung

Bei nicht eindeutiger Eingabe zeigt ein weiterer Druck auf <TAB> eine Liste von möglichen Alternativen an:

```
$ ls
aufgaben_linuxkurs_tag1.pdf  folien_linuxkurs_tag1.pdf
aufgaben_linuxkurs_tag2.pdf  folien_linuxkurs_tag2.pdf

$ file f<TAB>
$ file folien_linuxkurs_tag<TAB><TAB>
folien_linuxkurs_tag1.pdf  folien_linuxkurs_tag2.pdf
$ file folien_linuxkurs_tag2<TAB>
$ file folien_linuxkurs_tag2.pdf
folien_linuxkurs_tag2.pdf: PDF document, version 1.4
```

- ▶ Mit Cursortasten hoch/runter durch letzte Befehle bewegen
- ▶ `Ctrl-R` liefert den Modus „reverse-i-search“.
- ▶ Tippt man nun den Teil eines Befehls ein, erscheint der zuletzt benutzte Befehl, der diesen Teil enthält.
- ▶ Durch nochmaliges Drücken von `Ctrl-R` kann man durch mögliche Befehle scrollen.
- ▶ Hat man gefunden, was man sucht, kann man den Befehl noch beliebig editieren (Pfeiltaste zur Navigation) und dann ausführen.
- ▶ ... sehr praktisch für lange komplizierte Zeilen!

# Fahrt aufnehmen

## Copy & Paste in Terminals

**copy:** Den Text, den man kopieren will, einfach markieren...

**paste:** ... und an der gewünschten Stelle mit einem Klick auf das Mausrad (oder mit `Shift-Insert`) einfügen.

Viele Terminals unterstützen auch `Ctrl+Shift+C` und `Ctrl+Shift+V`.

# Elementare Befehle

manpages – das Hilfesystem unter Unix

## Typische Verwendung

```
man <Befehl>
```

### man echo

```
ECHO(1)                                User Commands                                ECHO(1)
```

#### NAME

```
echo - display a line of text
```

#### SYNOPSIS

```
echo [OPTION]... [STRING]...
```

#### DESCRIPTION

```
Echo the STRING(s) to standard output.
```

```
-n      do not output the trailing newline
```

## Die wichtigsten Tasten

- ▶ **Scrollen (zeilenweise)**: Pfeiltaste hoch/runter
- ▶ **Scrollen (seitenweise)**: Bild auf/ab
- ▶ **Suchen**: `/suchbegriff<ENTER>`
- ▶ **Nächster Treffer**: `n`
- ▶ **Vorheriger Treffer**: `N`
- ▶ **Beenden**: `q`

Tipp: Auch andere (terminalbasierte) Programme wie `less` lassen sich so bedienen!

# Elementare Befehle

`mkdir`, `rmdir` – Verzeichnisse erstellen und entfernen

## `mkdir`

`mkdir foo` legt ein Verzeichnis `foo` im aktuellen Verzeichnis an

## `rmdir`

`rmdir foo` löscht das Verzeichnis `foo` aus dem aktuellen Verzeichnis (`foo` muss leer sein)

# Elementare Befehle

mv – Verschieben

## Aufbau

```
mv <Quelle> <Ziel>
```

## Beispiele

- `mv alt neu` – benennt die Datei `alt` in `neu` um (geht auch für Verzeichnisse)
- `mv foo dinge/` – verschiebt die Datei `foo` aus dem aktuellen Verzeichnis in das Verzeichnis `dinge`

# Elementare Befehle

cp – Kopieren

## Aufbau

```
cp <Quelle> <Ziel>
```

## Beispiele

- `cp bsp bspkopie` – kopiert die Datei `bsp` nach `bspkopie` (im aktuellen Verzeichnis)
- `cp bsp test/` – kopiert die Datei `bsp` in das Verzeichnis `test`
- `cp -v bsp test/` – ... mit Ausgabe der einzelnen Kopieraktionen
- `cp -r test/ test2` – erstellt eine Kopie des Verzeichnisses `test` mit dem Namen `test2`
- `cp -r /verz .` – erstellt eine Kopie des Verzeichnisses `/verz` im aktuellen Verzeichnis

# Elementare Befehle

## rm – Löschen

rm

rm löscht Dateien und Verzeichnisse

### Beispiele

- `rm foo.pdf` – löscht die Datei `foo.pdf`
- `rm -r Mails/` – löscht das Verzeichnis `Mails` und alle darin enthaltenen Dateien und Unterverzeichnisse
- `rm -rf wichtig/` – löscht das Verzeichnis `wichtig` mit allen darin enthaltenen Dateien und Unterverzeichnissen, ohne nachzufragen – auch falls diese schreibgeschützt sind!

### Achtung!

rm löscht **ohne** Nachfrage und **ohne** Umweg über den Papierkorb!

# Elementare Befehle

mv, cp, rm – interaktive Nachfrage

## Achtung!

mv und cp überschreiben Dateien mit dem gleichen Namen!

**-i**

- mv -i, cp -i – fragt jedes Mal interaktiv nach, wenn eine Datei überschrieben werden würde
- rm -i – fragt bei jeder Datei und jedem Verzeichnis nach, ob sie gelöscht werden sollen

# Elementare Befehle

## Anzeige von Textdateien

Zur Ausgabe von Textdateien gibt es den Befehl `cat`.

### Typische Verwendung

```
cat <Datei>
```

```
$ cat elementare-befehle.tex
\begin{frame}
\frametitle{manpages -- das Hilfesystem unter Unix}
...
```

# Elementare Befehle

## Anzeige von Textdateien (2)

Hilfe, so schnell kann ich nicht lesen!

### Wie kann ich die Anzeige verlangsamen?

`cat` gibt eingelesene Datei komplett aus, egal wie groß diese ist.  
Seitenweise Anzeige: `less`.

### Typische Verwendung

```
less <Datei>
```

### Achtung!

- ▶ `cat` und `less` können nur Textdateien sinnvoll anzeigen.
- ▶ Falls nach der Ausgabe einer Binärdatei nur noch seltsame Zeichen dargestellt werden, hilft der Befehl `reset`.

# Wildcards

```
$ ls
linuxkurs2022.aux linuxkurs2022.log linuxkurs2022.nav
linuxkurs2022.pdf linuxkurs2022.tex linuxkurs2022.toc
linuxkurs2023.aux linuxkurs2023.log linuxkurs2023.nav
linuxkurs2023.pdf linuxkurs2023.tex linuxkurs2023.toc
```

Wie werde ich nur die ganzen Dateien vom letzten Jahr los?

```
$ rm linuxkurs2022.aux linuxkurs2022.log linuxkurs2022.nav
...
```

Geht das nicht einfacher?!

Aber natürlich.

## Platzhalter

Die *bash* erlaubt den Einsatz von Platzhalterzeichen („Wildcards“).

- ▶ \* steht für beliebig viele (oder auch keine) Zeichen
- ▶ ? steht für genau ein Zeichen

Zurück zum Beispiel:

```
$ rm linuxkurs2022*
```

`linuxkurs2022*` steht demnach für alle Dateinamen, die mit `linuxkurs2022` beginnen:

```
linuxkurs2022* ~> linuxkurs2022.aux linuxkurs2022.log ...
```

## Platzhalter II

Es geht auch noch etwas komplizierter:

- ▶ `[123]` steht für genau eines der Zeichen zwischen den eckigen Klammern: 1 2 3
- ▶ `[!123]` steht für ein Zeichen, das nicht zwischen den Klammern steht: z.B. a 4 J \_
- ▶ `[a-d]` steht für ein Zeichen aus dem angegebenen Bereich: a b c d
- ▶ `{1,2,abc}` steht der Reihe nach für *alle* der angegebenen Strings (unabhängig davon, ob eine Datei mit dem Namen existiert)

# Wildcards

## Beispiele

```
$ ls  
hand sand band  
  
$ ls [hbr]and  
hand band
```

```
$ wget http://www.example.net/folien{0,1,2,3,4}.pdf
```

Lädt die Dateien folien0.pdf, folien1.pdf, ... vom Server herunter

```
$ pdftk folien*.pdf cat output allefolien.pdf
```

... und baut die heruntergeladenen Dateien folien0.pdf, folien1.pdf, folien2.pdf, ... zu einer großen PDF-Datei zusammen.

Der \*-Platzhalter bezieht sich nur auf nicht-versteckte Dateien!

```
$ ls -a
.      ..      .bash_history  a.txt  mein_bild.jpg
$ rm *
$ ls -a
.      ..      .bash_history
```

### Achtung!

`rm .*` würde `.` theoretisch zu `..` expandieren!  
(die meisten `rm`-Versionen überprüfen das allerdings intern)

`lpr`

`lpr` druckt ein PDF- bzw. PS-Dokument aus.

Beispiel

```
lpr -Pps1acipd foo.pdf - druckt die Datei foo.pdf auf dem  
Drucker im CIP1 aus
```

# Drucken im CIP-Pool

## Allgemeines

General Page Setup Page Handling

Printer	Location	Status
 Print to File		
 Print to LPR		

Command Line:

**Range**

All Pages

Current Page

Pages:

**Copies**

Copies:  -

Collate 

Reverse

# Drucken im CIP-Pool

## Druckernamen

### ⟨Drucker⟩ – Druckernamen

ps⟨Stockwerk⟩⟨Buchstabe⟩cip⟨Suffix⟩⟨Doppelseitig⟩

⟨Stockwerk⟩	in welchem der Drucker steht
⟨Buchstabe⟩	Unterscheidung der einzelnen Drucker
⟨Doppelseitig⟩	d – Duplex lange Seite t – Duplex kurze Seite weglassen – kein Duplex

Die Namen der Drucker sind auch am Gerät abzulesen.

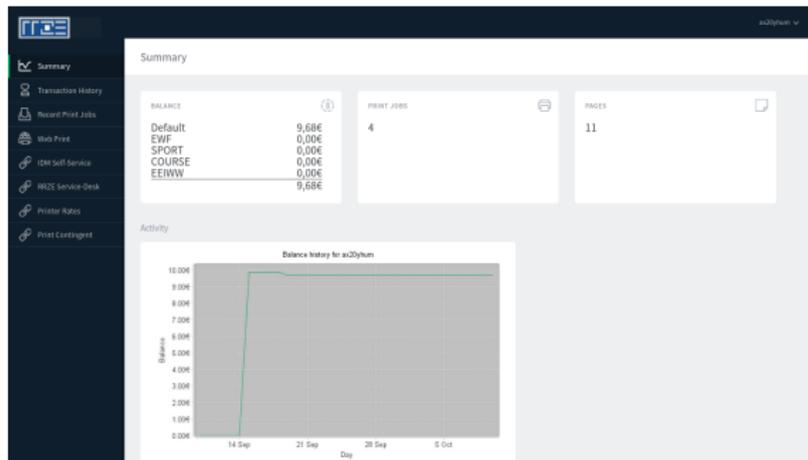
### Drucker ps2ccip

- ▶ Farbig drucken (teurer!); schwarz-weiß mit Suffix bw erzwingen
- ▶ Scannen (siehe Anleitung, die über dem Drucker an der Wand hängt)

# Drucken im CIP-Pool

## Druckerwarteschlange

Informationen zu Guthaben, Druckaufträgen (Erfolg/Fehler) können im RRZE-Druckerportal gefunden werden<sup>3</sup>:



Druckerguthaben kann an der Servicetheke (RRZE Raum 1.013) aufgeladen werden.

<sup>3</sup><https://fauprint.rrze.fau.de>

42

## Referenzen

- ▶ <https://en.flossmanuals.net/command-line>
- ▶ <https://fsi.cs.fau.de/linuxkurs>