



# Einführung in die Kommandozeile

## anhand der Bash auf Linux

Ralf Fischer

21.02.2007 / VHS-Kurs Einführung in die Kommandozeile  
der Linux User Schwabach e.V.



# Agenda

- 1 Einleitung
  - Wie sieht's aus?
  - Verzeichnisbaum
- 2 Getting started...
- 3 Editoren
- 4 Die Umgebung der Bash
- 5 Kommandos
  - alltägliche Befehle
  - Rechte und Dateisystem
  - Dateien Suchen
- 6 Shell für Fortgeschrittene
  - Job Control
  - Ausgabeumleitung



Wie sieht's aus?

# Agenda

- 1 **Einleitung**
  - **Wie sieht's aus?**
    - Verzeichnisbaum
- 2 Getting started...
- 3 Editoren
- 4 Die Umgebung der Bash
- 5 Kommandos
  - alltägliche Befehle
  - Rechte und Dateisystem
  - Dateien Suchen
- 6 Shell für Fortgeschrittene
  - Job Control
  - Ausgabeumleitung



Wie sieht's aus?

# Aufbau einer Shell

```
KeyChain 2.5.1; http://www.gentoo.org/pro/en/keychain/
Copyright 2002-2004 Gentoo Foundation; Distributed under the GPL

* Found existing ssh-agent (4631)
* Known ssh key: /home/makii/.ssh/id_dsa

Today is Sweetmorn, the 41st day of Chaos in the YOLD 3173

#-[ makii @ whitestar : ~ ]
#-[0]-> █
```

- Terminal
- Prompt
- Cursor



Wie sieht's aus?

# Aufbau einer Shell

```
KeyChain 2.5.1; http://www.gentoo.org/pro/en/keychain/
Copyright 2002-2004 Gentoo Foundation; Distributed under the GPL

* Found existing ssh-agent (4631)
* Known ssh key: /home/makii/.ssh/id_dsa

Today is Sweetmorn, the 41st day of Chaos in the YOLD 3173

#-[ makii @ whitestar : ~ ]
#-[0]-> █
```

- Terminal
- Prompt
- Cursor



Wie sieht's aus?

# Aufbau einer Shell

```
KeyChain 2.5.1; http://www.gentoo.org/pro/en/keychain/
Copyright 2002-2004 Gentoo Foundation; Distributed under the GPL

* Found existing ssh-agent (4631)
* Known ssh key: /home/makii/.ssh/id_dsa

Today is Sweetmorn, the 41st day of Chaos in the YOLD 3173

#-[ makii @ whitestar : ~ ]
#-[0]-> █
```

- Terminal
- Prompt
- Cursor



Wie sieht's aus?

# Und was mach ich damit?

Im Prinzip alles was man auch mit einer GUI kann:

- Arbeiten
- Briefe schreiben
- Emailen
- Surfen
- Administrieren
- **Und viel mehr!.**



Wie sieht's aus?

# Und was mach ich damit?

Im Prinzip alles was man auch mit einer GUI kann:

- Arbeiten
- Briefe schreiben
- Emails
- Surfen
- Administrieren
- **Und viel mehr!.**



Wie sieht's aus?

# Und was mach ich damit?

Im Prinzip alles was man auch mit einer GUI kann:

- Arbeiten
- Briefe schreiben
- Emailen
- Surfen
- Administrieren
- **Und viel mehr!.**



Wie sieht's aus?

# Und was mach ich damit?

Im Prinzip alles was man auch mit einer GUI kann:

- Arbeiten
- Briefe schreiben
- Emailen
- Surfen
- Administrieren
- Und viel mehr!.



Wie sieht's aus?

# Und was mach ich damit?

Im Prinzip alles was man auch mit einer GUI kann:

- Arbeiten
- Briefe schreiben
- Emailen
- Surfen
- Administrieren
- Und viel mehr!.



Wie sieht's aus?

# Und was mach ich damit?

Im Prinzip alles was man auch mit einer GUI kann:

- Arbeiten
- Briefe schreiben
- Emailen
- Surfen
- Administrieren
- Und viel mehr!.



Wie sieht's aus?

# Und was mach ich damit?

Im Prinzip alles was man auch mit einer GUI kann:

- Arbeiten
- Briefe schreiben
- Emailen
- Surfen
- Administrieren
- Und viel mehr!.



Wie sieht's aus?

# Und was mach ich damit?

Im Prinzip alles was man auch mit einer GUI kann:

- Arbeiten
- Briefe schreiben
- Emailen
- Surfen
- Administrieren
- **Und viel mehr!.**



# Agenda

- 1 **Einleitung**
  - Wie sieht's aus?
  - **Verzeichnisbaum**
- 2 Getting started...
- 3 Editoren
- 4 Die Umgebung der Bash
- 5 Kommandos
  - alltägliche Befehle
  - Rechte und Dateisystem
  - Dateien Suchen
- 6 Shell für Fortgeschrittene
  - Job Control
  - Ausgabeumleitung



# Ein Baum statt Laufwerke (1)

```
/ <root> # Wurzelverzeichnis
|-bin/      # Programme
|-boot/     # Dateien fuer den Bootloader
|  `--vmlinuz-2.6.16-1-686 # z.B. Kernel
|-dev/     # Geraetedateien
|-etc/     # Konfigurationen
|-home/    # Heimatverzeichnisse der Benutzer
|  `--johndoe/ # $HOME von Nutzer 'johndoe'
|-lib/     # Bibliotheken und Kern-Module
|-media/   # Wechselmedien
|-mnt/     # Andere, z.B. entfernte Medien
|-opt/     # Optionale Software
|-proc/    # Schnittstelle zum Kern
|-root/    # Home des Superusers
|-sbin/    # Administrationsprogramme
|-srv/     # Inhalte fuer Serverprogramme
```



## Ein Baum statt Laufwerke (2)

```
/ <root>
<...>
|-sys/           # Schnittstelle zur Hardware
|-tmp/          # Temporaerer Speicher
|-usr/          # Benutzerprogramme- und Daten
|  |-bin/       # Benutzerprogramme
|  |-share/
|  |  `-'doc/   # Dokumentation !
|  `-'X11/      # Graphische Benutzeroberflaeche
|    `-'bin
`-var/          # Variable Daten
```



# Hilfe zur Selbsthilfe!

## Die wichtigsten Kommandos zuerst: Hilfe!

- Parameter `-help`, `-h` zeigt meist Kurzhilfe zu Programmen
- `man` zeigt manpages zu Programmen an,
  - zb. `man ls`
- `info` zeigt Texinfo Anleitungen an,
  - zb. `info coreutils ls`
- `/usr/share/doc` Zentraler Ort für die Ablage von zusätzlicher Dokumentation.
  - Meist Text (gzipped) oder HTML
- Bei Ihrer lokalen LUG um die Ecke.
- ... und natürlich bei der Suchmaschine ihrer Wahl!

# Hilfe zur Selbsthilfe!

## Die wichtigsten Kommandos zuerst: Hilfe!

- **Parameter `-help`, `-h` zeigt meist Kurzhilfe zu Programmen**
- `man` zeigt manpages zu Programmen an,
  - `zb. man ls`
- `info` zeigt Texinfo Anleitungen an,
  - `zb. info coreutils ls`
- `/usr/share/doc` Zentraler Ort für die Ablage von zusätzlicher Dokumentation.
  - Meist Text (gzipped) oder HTML
- Bei Ihrer lokalen LUG um die Ecke.
- ... und natürlich bei der Suchmaschine ihrer Wahl!



# Hilfe zur Selbsthilfe!

## Die wichtigsten Kommandos zuerst: Hilfe!

- Parameter `-help`, `-h` zeigt meist Kurzhilfe zu Programmen
- `man` zeigt manpages zu Programmen an,
  - zb. `man ls`
- `info` zeigt Texinfo Anleitungen an,
  - zb. `info coreutils ls`
- `/usr/share/doc` Zentraler Ort für die Ablage von zusätzlicher Dokumentation.
  - Meist Text (gzipped) oder HTML
- Bei Ihrer lokalen LUG um die Ecke.
- ... und natürlich bei der Suchmaschine ihrer Wahl!



# Hilfe zur Selbsthilfe!

## Die wichtigsten Kommandos zuerst: Hilfe!

- Parameter `-help`, `-h` zeigt meist Kurzhilfe zu Programmen
- `man` zeigt manpages zu Programmen an,
  - zb. `man ls`
- `info` zeigt Texinfo Anleitungen an,
  - zb. `info coreutils ls`
- `/usr/share/doc` Zentraler Ort für die Ablage von zusätzlicher Dokumentation.
  - Meist Text (gzipped) oder HTML
- Bei Ihrer lokalen LUG um die Ecke.
- ... und natürlich bei der Suchmaschine ihrer Wahl!

# Hilfe zur Selbsthilfe!

## Die wichtigsten Kommandos zuerst: Hilfe!

- Parameter `-help`, `-h` zeigt meist Kurzhilfe zu Programmen
- `man` zeigt manpages zu Programmen an,
  - zb. `man ls`
- `info` zeigt Texinfo Anleitungen an,
  - zb. `info coreutils ls`
- `/usr/share/doc` Zentraler Ort für die Ablage von zusätzlicher Dokumentation.
  - Meist Text (gzipped) oder HTML
- Bei Ihrer lokalen LUG um die Ecke.
- ... und natürlich bei der Suchmaschine ihrer Wahl!



# Hilfe zur Selbsthilfe!

## Die wichtigsten Kommandos zuerst: Hilfe!

- Parameter `-help`, `-h` zeigt meist Kurzhilfe zu Programmen
- `man` zeigt manpages zu Programmen an,
  - zb. `man ls`
- `info` zeigt Texinfo Anleitungen an,
  - zb. `info coreutils ls`
- `/usr/share/doc` Zentraler Ort für die Ablage von zusätzlicher Dokumentation.
  - Meist Text (gzipped) oder HTML
- Bei Ihrer lokalen LUG um die Ecke.
- ... und natürlich bei der Suchmaschine ihrer Wahl!



# Hilfe zur Selbsthilfe!

## Die wichtigsten Kommandos zuerst: Hilfe!

- Parameter `-help`, `-h` zeigt meist Kurzhilfe zu Programmen
- `man` zeigt manpages zu Programmen an,
  - zb. `man ls`
- `info` zeigt Texinfo Anleitungen an,
  - zb. `info coreutils ls`
- `/usr/share/doc` Zentraler Ort für die Ablage von zusätzlicher Dokumentation.
  - Meist Text (gzipped) oder HTML
- Bei Ihrer lokalen LUG um die Ecke.
- ... und natürlich bei der Suchmaschine ihrer Wahl!

# Editoren (1)

Es gibt unterschiedlichste Konsolen-Editoren unter Linux. Hier eine kurze Auswahl der bekanntesten

- `vim` VI - iMproved
  - Nachfolger vom UNIX-`vi`
  - Erweiterbar durch eigene Scriptsprache
  - Hat unterschiedliche Modi
    - `command`
    - `edit`
    - `visual`
- `emacs` - Editor MACroS
  - Leicht erweiterbar durch LISP
  - Macros für alles mögliche vorhanden, zb.
    - Email-Client
    - Browser
    - MP3-Player



# Editoren (1)

Es gibt unterschiedlichste Konsolen-Editoren unter Linux. Hier eine kurze Auswahl der bekanntesten

- `vim` VI - iMproved
  - Nachfolger vom UNIX-`vi`
  - Erweiterbar durch eigene Scriptsprache
  - Hat unterschiedliche Modi
    - `command`
    - `edit`
    - `visual`
- `emacs` - Editor MACroS
  - Leicht erweiterbar durch LISP
  - Macros für alles mögliche vorhanden, zb.
    - Email-Client
    - Browser
    - MP3-Player



# Editoren (1)

Es gibt unterschiedlichste Konsolen-Editoren unter Linux. Hier eine kurze Auswahl der bekanntesten

- vim VI - iMproved
  - Nachfolger vom UNIX-vi
  - Erweiterbar durch eigene Scriptsprache
  - Hat unterschiedliche Modi
    - command
    - edit
    - visual
- emacs - Editor MACroS
  - Leicht erweiterbar durch LISP
  - Macros für alles mögliche vorhanden, zb.
    - Email-Client
    - Browser
    - MP3-Player



# Editoren (1)

Es gibt unterschiedlichste Konsolen-Editoren unter Linux. Hier eine kurze Auswahl der bekanntesten

- vim VI - iMproved
  - Nachfolger vom UNIX-vi
  - Erweiterbar durch eigene Scriptsprache
  - Hat unterschiedliche Modi
    - command
    - edit
    - visual
- emacs - Editor MACroS
  - Leicht erweiterbar durch LISP
  - Macros für alles mögliche vorhanden, zb.
    - Email-Client
    - Browser
    - MP3-Player



# Editoren (1)

Es gibt unterschiedlichste Konsolen-Editoren unter Linux. Hier eine kurze Auswahl der bekanntesten

- `vim` VI - iMproved
  - Nachfolger vom UNIX-`vi`
  - Erweiterbar durch eigene Scriptsprache
  - Hat unterschiedliche Modi
    - `command`
    - `edit`
    - `visual`
- `emacs` - Editor MACroS
  - Leicht erweiterbar durch LISP
  - Macros für alles mögliche vorhanden, zb.
    - Email-Client
    - Browser
    - MP3-Player

# Editoren (1)

Es gibt unterschiedlichste Konsolen-Editoren unter Linux. Hier eine kurze Auswahl der bekanntesten

- `vim` VI - iMproved
  - Nachfolger vom UNIX-`vi`
  - Erweiterbar durch eigene Scriptsprache
  - Hat unterschiedliche Modi
    - `command`
    - `edit`
    - `visual`
- `emacs` - Editor MACroS
  - Leicht erweiterbar durch LISP
  - Macros für alles mögliche vorhanden, zb.
    - Email-Client
    - Browser
    - MP3-Player



# Editoren (1)

Es gibt unterschiedlichste Konsolen-Editoren unter Linux. Hier eine kurze Auswahl der bekanntesten

- vim VI - iMproved
  - Nachfolger vom UNIX-vi
  - Erweiterbar durch eigene Scriptsprache
  - Hat unterschiedliche Modi
    - command
    - edit
    - visual
- emacs - Editor MACroS
  - Leicht erweiterbar durch LISP
  - Macros für alles mögliche vorhanden, zb.
    - Email-Client
    - Browser
    - MP3-Player



# Editoren (1)

Es gibt unterschiedlichste Konsolen-Editoren unter Linux. Hier eine kurze Auswahl der bekanntesten

- vim VI - iMproved
  - Nachfolger vom UNIX-vi
  - Erweiterbar durch eigene Scriptsprache
  - Hat unterschiedliche Modi
    - command
    - edit
    - visual
- emacs - Editor MACroS
  - Leicht erweiterbar durch LISP
  - Macros für alles mögliche vorhanden, zb.
    - Email-Client
    - Browser
    - MP3-Player



# Editoren (2)

- nano - Nano's ANOther editor
  - Designerter Nachfolger von Pico
  - Recht intuitiv für Umsteiger
    - Einfache Tastenkombinationen
    - Gute Eingebaute Hilfefunktion



# Editoren (2)

- nano - Nano's ANOther editor
  - Designerter Nachfolger von Pico
  - Recht intuitiv für Umsteiger
    - Einfache Tastenkombinationen
    - Gute Eingebaute Hilfefunktion



## Editoren (2)

- nano - Nano's ANOther editor
  - Designerter Nachfolger von Pico
  - Recht intuitiv für Umsteiger
    - Einfache Tastenkombinationen
    - Gute Eingebaute Hilfefunktion



# Editoren (2)

- nano - Nano's ANOther editor
  - Designerter Nachfolger von Pico
  - Recht intuitiv für Umsteiger
    - Einfache Tastenkombinationen
    - Gute Eingebaute Hilfefunktion



## Editoren (2)

- nano - Nano's ANOther editor
  - Designerter Nachfolger von Pico
  - Recht intuitiv für Umsteiger
    - Einfache Tastenkombinationen
    - Gute Eingebaute Hilfefunktion



# Das Kurzzeitgedächtnis

- Hier speichern Shell und Programme Informationen
- Ausgeben den Umgebung mit `env`
- Beispiele:
  - `OLDPWD`: hier steht das "letzte" Verzeichnis, kann mit `cd -` wieder angesprochen werden
  - `HOME`: Das Heimatverzeichnis den Benutzers
  - `EDITOR`: Der Default-Editor des Benutzers
  - `PATH`: Der Programmpfad in dem ausführbare Programme gesucht werden
- `export` setzt Variablen in die Umgebung
- `$` referenziert Variablen: `echo $PATH`
- `unset` löscht Variablen aus der Umgebung



# Das Kurzzeitgedächtnis

- Hier speichern Shell und Programme Informationen
- Ausgeben den Umgebung mit `env`
- Beispiele:
  - `OLDPWD`: hier steht das "letzte" Verzeichnis, kann mit `cd -` wieder angesprochen werden
  - `HOME`: Das Heimatverzeichnis den Benutzers
  - `EDITOR`: Der Default-Editor des Benutzers
  - `PATH`: Der Programmpfad in dem ausführbare Programme gesucht werden
- `export` setzt Variablen in die Umgebung
- `$` referenziert Variablen: `echo $PATH`
- `unset` löscht Variablen aus der Umgebung



# Das Kurzzeitgedächtnis

- Hier speichern Shell und Programme Informationen
- Ausgeben den Umgebung mit `env`
- Beispiele:
  - `OLDPWD`: hier steht das "letzte" Verzeichnis, kann mit `cd -` wieder angesprungen werden
  - `HOME`: Das Heimatverzeichnis den Benutzers
  - `EDITOR`: Der Default-Editor des Benutzers
  - `PATH`: Der Programmpfad in dem ausführbare Programme gesucht werden
- `export` setzt Variablen in die Umgebung
- `$` referenziert Variablen: `echo $PATH`
- `unset` löscht Variablen aus der Umgebung



# Das Kurzzeitgedächtnis

- Hier speichern Shell und Programme Informationen
- Ausgeben den Umgebung mit `env`
- Beispiele:
  - `OLDPWD`: hier steht das "letzte" Verzeichnis, kann mit `cd -` wieder angesprungen werden
  - `HOME`: Das Heimatverzeichnis den Benutzers
  - `EDITOR`: Der Default-Editor des Benutzers
  - `PATH`: Der Programmpfad in dem ausführbare Programme gesucht werden
- `export` setzt Variablen in die Umgebung
- `$` referenziert Variablen: `echo $PATH`
- `unset` löscht Variablen aus der Umgebung

# Das Kurzzeitgedächtnis

- Hier speichern Shell und Programme Informationen
- Ausgeben den Umgebung mit `env`
- Beispiele:
  - `OLDPWD`: hier steht das "letzte" Verzeichnis, kann mit `cd -` wieder angesprungen werden
  - `HOME`: Das Heimatverzeichnis den Benutzers
  - `EDITOR`: Der Default-Editor des Benutzers
  - `PATH`: Der Programmpfad in dem ausführbare Programme gesucht werden
- `export` setzt Variablen in die Umgebung
- `$` referenziert Variablen: `echo $PATH`
- `unset` löscht Variablen aus der Umgebung



# Das Kurzzeitgedächtnis

- Hier speichern Shell und Programme Informationen
- Ausgeben den Umgebung mit `env`
- Beispiele:
  - `OLDPWD`: hier steht das "letzte" Verzeichnis, kann mit `cd -` wieder angesprungen werden
  - `HOME`: Das Heimatverzeichnis den Benutzers
  - `EDITOR`: Der Default-Editor des Benutzers
  - `PATH`: Der Programmpfad in dem ausführbare Programme gesucht werden
- `export` setzt Variablen in die Umgebung
- `$` referenziert Variablen: `echo $PATH`
- `unset` löscht Variablen aus der Umgebung



# Das Kurzzeitgedächtnis

- Hier speichern Shell und Programme Informationen
- Ausgeben den Umgebung mit `env`
- Beispiele:
  - `OLDPWD`: hier steht das "letzte" Verzeichnis, kann mit `cd -` wieder angesprungen werden
  - `HOME`: Das Heimatverzeichnis den Benutzers
  - `EDITOR`: Der Default-Editor des Benutzers
  - `PATH`: Der Programmpfad in dem ausführbare Programme gesucht werden
- `export` setzt Variablen in die Umgebung
- `$` referenziert Variablen: `echo $PATH`
- `unset` löscht Variablen aus der Umgebung



# Das Kurzzeitgedächtnis

- Hier speichern Shell und Programme Informationen
- Ausgeben den Umgebung mit `env`
- Beispiele:
  - `OLDPWD`: hier steht das "letzte" Verzeichnis, kann mit `cd -` wieder angesprungen werden
  - `HOME`: Das Heimatverzeichnis den Benutzers
  - `EDITOR`: Der Default-Editor des Benutzers
  - `PATH`: Der Programmpfad in dem ausführbare Programme gesucht werden
- `export` setzt Variablen in die Umgebung
- `$` referenziert Variablen: `echo $PATH`
- `unset` löscht Variablen aus der Umgebung



# Das Kurzzeitgedächtnis

- Hier speichern Shell und Programme Informationen
- Ausgeben den Umgebung mit `env`
- Beispiele:
  - `OLDPWD`: hier steht das "letzte" Verzeichnis, kann mit `cd -` wieder angesprungen werden
  - `HOME`: Das Heimatverzeichnis den Benutzers
  - `EDITOR`: Der Default-Editor des Benutzers
  - `PATH`: Der Programmpfad in dem ausführbare Programme gesucht werden
- `export` setzt Variablen in die Umgebung
- `$` referenziert Variablen: `echo $PATH`
- `unset` löscht Variablen aus der Umgebung

# Die Kurzwahltasten: alias

- Die Shell kann Aliase für oft verwendete Befehle verwalten
- `alias` erstellt Aliase und zeigt sie an
- `unalias` löscht sie wieder

```
#-[ makii @ whitestar : ~/files/speeches/linux-commandline ]
#-[0]-> alias foobar='ls -lh'

#-[ makii @ whitestar : ~/files/speeches/linux-commandline ]
#-[0]-> alias | grep foo
alias foobar='ls -lh'

#-[ makii @ whitestar : ~/files/speeches/linux-commandline ]
#-[0]-> foobar
total 1.5M
-rw-r--r--  1 makii makii  27K 2007-02-10 14:03 Eterm.jpg
-rw-r--r--  1 makii makii  7.8K 2007-02-21 05:07 Linux_on_the_Shell.aux
-rw-r--r--  1 makii makii  28K 2007-02-21 05:07 Linux_on_the_Shell.log
-rw-r--r--  1 makii makii  4.7K 2007-02-21 05:07 Linux_on_the_Shell.nav
<snip />
```



# Die Kurzwahltasten: alias

- Die Shell kann Aliase für oft verwendete Befehle verwalten
- `alias` erstellt Aliase und zeigt sie an
- `unalias` löscht sie wieder

```
#-[ makii @ whitestar : ~/files/speeches/linux-commandline ]
#-[0]-> alias foobar='ls -lh'
```

```
#-[ makii @ whitestar : ~/files/speeches/linux-commandline ]
#-[0]-> alias | grep foo
alias foobar='ls -lh'
```

```
#-[ makii @ whitestar : ~/files/speeches/linux-commandline ]
#-[0]-> foobar
```

```
total 1.5M
-rw-r--r--  1 makii makii  27K 2007-02-10 14:03 Eterm.jpg
-rw-r--r--  1 makii makii  7.8K 2007-02-21 05:07 Linux_on_the_Shell.aux
-rw-r--r--  1 makii makii  28K 2007-02-21 05:07 Linux_on_the_Shell.log
-rw-r--r--  1 makii makii  4.7K 2007-02-21 05:07 Linux_on_the_Shell.nav
<snip />
```

# Die Kurzwahltasten: alias

- Die Shell kann Aliase für oft verwendete Befehle verwalten
- `alias` erstellt Aliase und zeigt sie an
- `unalias` löscht sie wieder

```
#-[ makii @ whitestar : ~/files/speeches/linux-commandline ]  
#-[0]-> alias foobar='ls -lh'
```

```
#-[ makii @ whitestar : ~/files/speeches/linux-commandline ]  
#-[0]-> alias | grep foo  
alias foobar='ls -lh'
```

```
#-[ makii @ whitestar : ~/files/speeches/linux-commandline ]  
#-[0]-> foobar
```

```
total 1.5M  
-rw-r--r--  1 makii makii  27K 2007-02-10 14:03 Eterm.jpg  
-rw-r--r--  1 makii makii  7.8K 2007-02-21 05:07 Linux_on_the_Shell.aux  
-rw-r--r--  1 makii makii  28K 2007-02-21 05:07 Linux_on_the_Shell.log  
-rw-r--r--  1 makii makii  4.7K 2007-02-21 05:07 Linux_on_the_Shell.nav  
<snip />
```



# Die Kurzwahltasten: alias

- Die Shell kann Aliase für oft verwendete Befehle verwalten
- `alias` erstellt Aliase und zeigt sie an
- `unalias` löscht sie wieder

```
#-[ makii @ whitestar : ~/files/speeches/linux-commandline ]  
#-[0]-> alias foobar='ls -lh'
```

```
#-[ makii @ whitestar : ~/files/speeches/linux-commandline ]  
#-[0]-> alias | grep foo  
alias foobar='ls -lh'
```

```
#-[ makii @ whitestar : ~/files/speeches/linux-commandline ]  
#-[0]-> foobar  
total 1.5M  
-rw-r--r--  1 makii makii  27K 2007-02-10 14:03 Eterm.jpg  
-rw-r--r--  1 makii makii  7.8K 2007-02-21 05:07 Linux_on_the_Shell.aux  
-rw-r--r--  1 makii makii  28K 2007-02-21 05:07 Linux_on_the_Shell.log  
-rw-r--r--  1 makii makii  4.7K 2007-02-21 05:07 Linux_on_the_Shell.nav  
<snip />
```



# Startskript - Jetzt bringt das auch was

- Beim Start der interaktiven Shell wird ein Startup-Skript ausgeführt
- Bei Bash:
  - `/etc/bash.bashrc` Globale Konfiguration (Debian, Suse kann abweichen)
  - `$HOME/.bashrc` Per-User Konfiguration
- Kommandos werden beim Start einer neuen Shell ausgeführt und in die Umgebung integriert (`gesourced`)
- Nützlich für
  - Aliase
  - Umgebungsvariablen
  - Programme, zb. `keychain`, Datum, reminders...



# Startskript - Jetzt bringt das auch was

- Beim Start der interaktiven Shell wird ein Startup-Skript ausgeführt
- Bei Bash:
  - `/etc/bash.bashrc` Globale Konfiguration (Debian, Suse kann abweichen)
  - `$HOME/.bashrc` Per-User Konfiguration
- Kommandos werden beim Start einer neuen Shell ausgeführt und in die Umgebung integriert (`gesourced`)
- Nützlich für
  - Aliase
  - Umgebungsvariablen
  - Programme, zb. `keychain`, Datum, reminders...



# Startskript - Jetzt bringt das auch was

- Beim Start der interaktiven Shell wird ein Startup-Skript ausgeführt
- Bei Bash:
  - `/etc/bash.bashrc` Globale Konfiguration (Debian, Suse kann abweichen)
  - `$HOME/.bashrc` Per-User Konfiguration
- Kommandos werden beim Start einer neuen Shell ausgeführt und in die Umgebung integriert (`gesourced`)
- Nützlich für
  - Aliase
  - Umgebungsvariablen
  - Programme, zb. `keychain`, Datum, reminders...



# Startskript - Jetzt bringt das auch was

- Beim Start der interaktiven Shell wird ein Startup-Skript ausgeführt
- Bei Bash:
  - `/etc/bash.bashrc` Globale Konfiguration (Debian, Suse kann abweichen)
  - `$HOME/.bashrc` Per-User Konfiguration
- Kommandos werden beim Start einer neuen Shell ausgeführt und in die Umgebung integriert (`gesourced`)
- Nützlich für
  - Aliase
  - Umgebungsvariablen
  - Programme, zb. `keychain`, Datum, reminders...



# Startskript - Jetzt bringt das auch was

- Beim Start der interaktiven Shell wird ein Startup-Skript ausgeführt
- Bei Bash:
  - `/etc/bash.bashrc` Globale Konfiguration (Debian, Suse kann abweichen)
  - `$HOME/.bashrc` Per-User Konfiguration
- Kommandos werden beim Start einer neuen Shell ausgeführt und in die Umgebung integriert (`gesourced`)
- Nützlich für
  - Aliase
  - Umgebungsvariablen
  - Programme, zb. `keychain`, Datum, reminders...



# Startskript - Jetzt bringt das auch was

- Beim Start der interaktiven Shell wird ein Startup-Skript ausgeführt
- Bei Bash:
  - `/etc/bash.bashrc` Globale Konfiguration (Debian, Suse kann abweichen)
  - `$HOME/.bashrc` Per-User Konfiguration
- Kommandos werden beim Start einer neuen Shell ausgeführt und in die Umgebung integriert (`gesourced`)
- Nützlich für
  - Aliase
  - Umgebungsvariablen
  - Programme, zb. `keychain`, Datum, reminders...



# Startskript - Jetzt bringt das auch was

- Beim Start der interaktiven Shell wird ein Startup-Skript ausgeführt
- Bei Bash:
  - `/etc/bash.bashrc` Globale Konfiguration (Debian, Suse kann abweichen)
  - `$HOME/.bashrc` Per-User Konfiguration
- Kommandos werden beim Start einer neuen Shell ausgeführt und in die Umgebung integriert (`gesourced`)
- Nützlich für
  - Aliase
  - Umgebungsvariablen
  - Programme, zb. `keychain`, Datum, reminders...



# Agenda

- 1 Einleitung
  - Wie sieht's aus?
  - Verzeichnisbaum
- 2 Getting started...
- 3 Editoren
- 4 Die Umgebung der Bash
- 5 Kommandos**
  - **alltägliche Befehle**
  - Rechte und Dateisystem
  - Dateien Suchen
- 6 Shell für Fortgeschrittene
  - Job Control
  - Ausgabeumleitung

# Umgang mit Dateien

## Was kann ich mit den Buchstaben jetzt machen?

- `ls` zeigt Dateien an
- `cd` wechselt das aktuelle Verzeichnis
- `file` errät den Dateityp
- `touch` erstellt eine leere Datei
- `rm` löscht Dateien, `rmdir` Verzeichnisse
- `grep` sucht in Dateien
- `cat` zeigt Dateien an ("catalog")
- `more` oder `less` geben Dateien seitenweise aus
- `head` zeigt den Dateianfang
- `tail` zeigt das Dateiende



# Umgang mit Dateien

## Was kann ich mit den Buchstaben jetzt machen?

- `ls` zeigt Dateien an
- `cd` wechselt das aktuelle Verzeichnis
- `file` errät den Dateityp
- `touch` erstellt eine leere Datei
- `rm` löscht Dateien, `rmdir` Verzeichnisse
- `grep` sucht in Dateien
- `cat` zeigt Dateien an ("catalog")
- `more` oder `less` geben Dateien seitenweise aus
- `head` zeigt den Dateianfang
- `tail` zeigt das Dateiende

# Umgang mit Dateien

Was kann ich mit den Buchstaben jetzt machen?

- `ls` zeigt Dateien an
- `cd` wechselt das aktuelle Verzeichnis
- `file` errät den Dateityp
- `touch` erstellt eine leere Datei
- `rm` löscht Dateien, `rmdir` Verzeichnisse
- `grep` sucht in Dateien
- `cat` zeigt Dateien an ("catalog")
- `more` oder `less` geben Dateien seitenweise aus
- `head` zeigt den Dateianfang
- `tail` zeigt das Dateiende

# Umgang mit Dateien

Was kann ich mit den Buchstaben jetzt machen?

- `ls` zeigt Dateien an
- `cd` wechselt das aktuelle Verzeichnis
- `file` errät den Dateityp
- `touch` erstellt eine leere Datei
- `rm` löscht Dateien, `rmdir` Verzeichnisse
- `grep` sucht in Dateien
- `cat` zeigt Dateien an ("catalog")
- `more` oder `less` geben Dateien seitenweise aus
- `head` zeigt den Dateianfang
- `tail` zeigt das Dateiende

# Umgang mit Dateien

Was kann ich mit den Buchstaben jetzt machen?

- `ls` zeigt Dateien an
- `cd` wechselt das aktuelle Verzeichnis
- `file` errät den Dateityp
- `touch` erstellt eine leere Datei
- `rm` löscht Dateien, `rmdir` Verzeichnisse
- `grep` sucht in Dateien
- `cat` zeigt Dateien an ("catalog")
- `more` oder `less` geben Dateien seitenweise aus
- `head` zeigt den Dateianfang
- `tail` zeigt das Dateiende



# Umgang mit Dateien

Was kann ich mit den Buchstaben jetzt machen?

- `ls` zeigt Dateien an
- `cd` wechselt das aktuelle Verzeichnis
- `file` errät den Dateityp
- `touch` erstellt eine leere Datei
- `rm` löscht Dateien, `rmdir` Verzeichnisse
- `grep` sucht in Dateien
- `cat` zeigt Dateien an ("catalog")
- `more` oder `less` geben Dateien seitenweise aus
- `head` zeigt den Dateianfang
- `tail` zeigt das Dateiende

# Umgang mit Dateien

Was kann ich mit den Buchstaben jetzt machen?

- `ls` zeigt Dateien an
- `cd` wechselt das aktuelle Verzeichnis
- `file` errät den Dateityp
- `touch` erstellt eine leere Datei
- `rm` löscht Dateien, `rmdir` Verzeichnisse
- `grep` sucht in Dateien
- `cat` zeigt Dateien an ("catalog")
- `more` oder `less` geben Dateien seitenweise aus
- `head` zeigt den Dateianfang
- `tail` zeigt das Dateiende



# Umgang mit Dateien

Was kann ich mit den Buchstaben jetzt machen?

- `ls` zeigt Dateien an
- `cd` wechselt das aktuelle Verzeichnis
- `file` errät den Dateityp
- `touch` erstellt eine leere Datei
- `rm` löscht Dateien, `rmdir` Verzeichnisse
- `grep` sucht in Dateien
- `cat` zeigt Dateien an ("catalog")
- `more` oder `less` geben Dateien seitenweise aus
- `head` zeigt den Dateianfang
- `tail` zeigt das Dateiende



# Umgang mit Dateien

Was kann ich mit den Buchstaben jetzt machen?

- `ls` zeigt Dateien an
- `cd` wechselt das aktuelle Verzeichnis
- `file` errät den Dateityp
- `touch` erstellt eine leere Datei
- `rm` löscht Dateien, `rmdir` Verzeichnisse
- `grep` sucht in Dateien
- `cat` zeigt Dateien an ("catalog")
- `more` oder `less` geben Dateien seitenweise aus
- `head` zeigt den Dateianfang
- `tail` zeigt das Dateiende



# Umgang mit Dateien

Was kann ich mit den Buchstaben jetzt machen?

- `ls` zeigt Dateien an
- `cd` wechselt das aktuelle Verzeichnis
- `file` errät den Dateityp
- `touch` erstellt eine leere Datei
- `rm` löscht Dateien, `rmdir` Verzeichnisse
- `grep` sucht in Dateien
- `cat` zeigt Dateien an ("catalog")
- `more` oder `less` geben Dateien seitenweise aus
- `head` zeigt den Dateianfang
- `tail` zeigt das Dateiende



# Umgang mit Dateien

Was kann ich mit den Buchstaben jetzt machen?

- `ls` zeigt Dateien an
- `cd` wechselt das aktuelle Verzeichnis
- `file` errät den Dateityp
- `touch` erstellt eine leere Datei
- `rm` löscht Dateien, `rmdir` Verzeichnisse
- `grep` sucht in Dateien
- `cat` zeigt Dateien an ("catalog")
- `more` oder `less` geben Dateien seitenweise aus
- `head` zeigt den Dateianfang
- `tail` zeigt das Dateiende



# Wichtige Programme

- `ps` zeigt Prozesse auf dem System an
- `netstat` zeigt Netzwerkverbindungen an
- `who` zeigt eingeloggte Benutzer an
- `du` zeigt verwendeten Speicher an
- `df` zeigt freien Festspeicher an
- `free` zeigt freien Arbeitsspeicher an



# Wichtige Programme

- `ps` zeigt Prozesse auf dem System an
- `netstat` zeigt Netzwerkverbindungen an
- `who` zeigt eingeloggte Benutzer an
- `du` zeigt verwendeten Speicher an
- `df` zeigt freien Festspeicher an
- `free` zeigt freien Arbeitsspeicher an



# Wichtige Programme

- `ps` zeigt Prozesse auf dem System an
- `netstat` zeigt Netzwerkverbindungen an
- `who` zeigt eingeloggte Benutzer an
- `du` zeigt verwendeten Speicher an
- `df` zeigt freien Festspeicher an
- `free` zeigt freien Arbeitsspeicher an



# Wichtige Programme

- `ps` zeigt Prozesse auf dem System an
- `netstat` zeigt Netzwerkverbindungen an
- `who` zeigt eingeloggte Benutzer an
- `du` zeigt verwendeten Speicher an
- `df` zeigt freien Festspeicher an
- `free` zeigt freien Arbeitsspeicher an



# Wichtige Programme

- `ps` zeigt Prozesse auf dem System an
- `netstat` zeigt Netzwerkverbindungen an
- `who` zeigt eingeloggte Benutzer an
- `du` zeigt verwendeten Speicher an
- `df` zeigt freien Festspeicher an
- `free` zeigt freien Arbeitsspeicher an



# Wichtige Programme

- `ps` zeigt Prozesse auf dem System an
- `netstat` zeigt Netzwerkverbindungen an
- `who` zeigt eingeloggte Benutzer an
- `du` zeigt verwendeten Speicher an
- `df` zeigt freien Festspeicher an
- `free` zeigt freien Arbeitsspeicher an



# Wichtige Programme

- `ps` zeigt Prozesse auf dem System an
- `netstat` zeigt Netzwerkverbindungen an
- `who` zeigt eingeloggte Benutzer an
- `du` zeigt verwendeten Speicher an
- `df` zeigt freien Festspeicher an
- `free` zeigt freien Arbeitsspeicher an



# Administratorprogramme

- **kill** sendet Signale an Prozesse, z.B. TERM, QUIT
- `mount` bindet neue Partitionen in den Verzeichnisbaum ein
- `mkfs` erzeugt Dateisysteme
- `fdisk` zeigt Informationen zu Massenspeichern an
- `ifconfig` konfiguriert das Netzwerk-Interface
- `iwconfig` konfiguriert das Wireless-Interface
- `iwlist` zeigt Informationen zu Wireless-Netzwerken an
- `route` zeigt Routinginformationen an
- `lsmod` zeigt Kernmodule an
- `modprobe` lädt Kernmodule



# Administratorprogramme

- `kill` sendet Signale an Prozesse, z.B. `TERM`, `QUIT`
- `mount` bindet neue Partitionen in den Verzeichnisbaum ein
- `mkfs` erzeugt Dateisysteme
- `fdisk` zeigt Informationen zu Massenspeichern an
- `ifconfig` konfiguriert das Netzwerk-Interface
- `iwconfig` konfiguriert das Wireless-Interface
- `iwlist` zeigt Informationen zu Wireless-Netzwerken an
- `route` zeigt Routinginformationen an
- `lsmod` zeigt Kernmodule an
- `modprobe` lädt Kernmodule



# Administratorprogramme

- `kill` sendet Signale an Prozesse, z.B. `TERM`, `QUIT`
- `mount` bindet neue Partitionen in den Verzeichnisbaum ein
- `mkfs` erzeugt Dateisysteme
- `fdisk` zeigt Informationen zu Massenspeichern an
- `ifconfig` konfiguriert das Netzwerk-Interface
- `iwconfig` konfiguriert das Wireless-Interface
- `iwlist` zeigt Informationen zu Wireless-Netzwerken an
- `route` zeigt Routinginformationen an
- `lsmod` zeigt Kernmodule an
- `modprobe` lädt Kernmodule



# Administratorprogramme

- `kill` sendet Signale an Prozesse, z.B. `TERM`, `QUIT`
- `mount` bindet neue Partitionen in den Verzeichnisbaum ein
- `mkfs` erzeugt Dateisysteme
- `fdisk` zeigt Informationen zu Massenspeichern an
- `ifconfig` konfiguriert das Netzwerk-Interface
- `iwconfig` konfiguriert das Wireless-Interface
- `iwlist` zeigt Informationen zu Wireless-Netzwerken an
- `route` zeigt Routinginformationen an
- `lsmod` zeigt Kernmodule an
- `modprobe` lädt Kernmodule



# Administratorprogramme

- `kill` sendet Signale an Prozesse, z.B. `TERM`, `QUIT`
- `mount` bindet neue Partitionen in den Verzeichnisbaum ein
- `mkfs` erzeugt Dateisysteme
- `fdisk` zeigt Informationen zu Massenspeichern an
- `ifconfig` konfiguriert das Netzwerk-Interface
- `iwconfig` konfiguriert das Wireless-Interface
- `iwlist` zeigt Informationen zu Wireless-Netzwerken an
- `route` zeigt Routinginformationen an
- `lsmod` zeigt Kernmodule an
- `modprobe` lädt Kernmodule



# Administratorprogramme

- `kill` sendet Signale an Prozesse, z.B. `TERM`, `QUIT`
- `mount` bindet neue Partitionen in den Verzeichnisbaum ein
- `mkfs` erzeugt Dateisysteme
- `fdisk` zeigt Informationen zu Massenspeichern an
- `ifconfig` konfiguriert das Netzwerk-Interface
- `iwconfig` konfiguriert das Wireless-Interface
- `iwlist` zeigt Informationen zu Wireless-Netzwerken an
- `route` zeigt Routinginformationen an
- `lsmod` zeigt Kernmodule an
- `modprobe` lädt Kernmodule



# Administratorprogramme

- `kill` sendet Signale an Prozesse, z.B. `TERM`, `QUIT`
- `mount` bindet neue Partitionen in den Verzeichnisbaum ein
- `mkfs` erzeugt Dateisysteme
- `fdisk` zeigt Informationen zu Massenspeichern an
- `ifconfig` konfiguriert das Netzwerk-Interface
- `iwconfig` konfiguriert das Wireless-Interface
- `iwlist` zeigt Informationen zu Wireless-Netzwerken an
- `route` zeigt Routinginformationen an
- `lsmod` zeigt Kernmodule an
- `modprobe` lädt Kernmodule



# Administratorprogramme

- `kill` sendet Signale an Prozesse, z.B. `TERM`, `QUIT`
- `mount` bindet neue Partitionen in den Verzeichnisbaum ein
- `mkfs` erzeugt Dateisysteme
- `fdisk` zeigt Informationen zu Massenspeichern an
- `ifconfig` konfiguriert das Netzwerk-Interface
- `iwconfig` konfiguriert das Wireless-Interface
- `iwlist` zeigt Informationen zu Wireless-Netzwerken an
- `route` zeigt Routinginformationen an
- `lsmod` zeigt Kernmodule an
- `modprobe` lädt Kernmodule



# Administratorprogramme

- `kill` sendet Signale an Prozesse, z.B. `TERM`, `QUIT`
- `mount` bindet neue Partitionen in den Verzeichnisbaum ein
- `mkfs` erzeugt Dateisysteme
- `fdisk` zeigt Informationen zu Massenspeichern an
- `ifconfig` konfiguriert das Netzwerk-Interface
- `iwconfig` konfiguriert das Wireless-Interface
- `iwlist` zeigt Informationen zu Wireless-Netzwerken an
- `route` zeigt Routinginformationen an
- `lsmod` zeigt Kernmodule an
- `modprobe` lädt Kernmodule



# Administratorprogramme

- `kill` sendet Signale an Prozesse, z.B. `TERM`, `QUIT`
- `mount` bindet neue Partitionen in den Verzeichnisbaum ein
- `mkfs` erzeugt Dateisysteme
- `fdisk` zeigt Informationen zu Massenspeichern an
- `ifconfig` konfiguriert das Netzwerk-Interface
- `iwconfig` konfiguriert das Wireless-Interface
- `iwlist` zeigt Informationen zu Wireless-Netzwerken an
- `route` zeigt Routinginformationen an
- `lsmod` zeigt Kernmodule an
- `modprobe` lädt Kernmodule



# Unter Druck: Kompression

- **gzip**
  - schnell und leicht
  - gute Kompressionsrate
  - `gzip <file>`
  - `gzip -d <file> / gunzip <file>`
- **bzip2**
  - langsamer
  - sehr gute Kompressionsrate
  - `bzip2 <file>`
  - `bzip2 -d <file> / bunzip2 <file>`
- **zip und unzip gibt's auch**



# Unter Druck: Kompression

- `gzip`
  - schnell und leicht
  - gute Kompressionsrate
  - `gzip <file>`
  - `gzip -d <file> / gunzip <file>`
- `bzip2`
  - langsamer
  - sehr gute Kompressionsrate
  - `bzip2 <file>`
  - `bzip2 -d <file> / bunzip2 <file>`
- `zip` und `unzip` gibt's auch



# Unter Druck: Kompression

- `gzip`
  - schnell und leicht
  - gute Kompressionsrate
  - `gzip <file>`
  - `gzip -d <file> / gunzip <file>`
- `bzip2`
  - langsamer
  - sehr gute Kompressionsrate
  - `bzip2 <file>`
  - `bzip2 -d <file> / bunzip2 <file>`
- `zip` und `unzip` gibt's auch



# Unter Druck: Kompression

- `gzip`
  - schnell und leicht
  - gute Kompressionsrate
  - `gzip <file>`
  - `gzip -d <file> / gunzip <file>`
- `bzip2`
  - langsamer
  - sehr gute Kompressionsrate
  - `bzip2 <file>`
  - `bzip2 -d <file> / bunzip2 <file>`
- `zip` und `unzip` gibt's auch



# Unter Druck: Kompression

- `gzip`
  - schnell und leicht
  - gute Kompressionsrate
  - `gzip <file>`
  - `gzip -d <file> / gunzip <file>`
- `bzip2`
  - langsamer
  - sehr gute Kompressionsrate
  - `bzip2 <file>`
  - `bzip2 -d <file> / bunzip2 <file>`
- `zip` und `unzip` gibt's auch

## ... und in den Koffer packe ich: Archivierung

- tar - Tape ARchiver
  - Archiviert Dateien
    - Stammt noch aus den Zeiten der Bandlaufwerke
    - `tar cf foobar.tar *` erzeugt Archiv
    - `tar tf foobar.tar *` zeigt Archivinhalt an
    - `tar xf foobar.tar *` extrahiert Archivinhalt
  - Integration mit Kompressionsprogrammen
    - `gzip: tar czf foobar.tar lala/ lulu/`
    - `bzip2: tar cjf foobar.tar lala/ lulu/`



## ... und in den Koffer packe ich: Archivierung

- tar - Tape ARchiver
  - Archiviert Dateien
  - Stammt noch aus den Zeiten der Bandlaufwerke
    - `tar cf foobar.tar *` erzeugt Archiv
    - `tar tf foobar.tar *` zeigt Archivinhalt an
    - `tar xf foobar.tar *` extrahiert Archivinhalt
  - Integration mit Kompressionsprogrammen
    - `gzip: tar czf foobar.tar lala/ lulu/`
    - `bzip2: tar cjf foobar.tar lala/ lulu/`

# ... und in den Koffer packe ich: Archivierung

- tar - Tape ARchiver
  - Archiviert Dateien
  - Stammt noch aus den Zeiten der Bandlaufwerke
  - `tar cf foobar.tar *` erzeugt Archiv
  - `tar tf foobar.tar *` zeigt Archivinhalt an
  - `tar xf foobar.tar *` extrahiert Archivinhalt
- Integration mit Kompressionsprogrammen
  - `gzip: tar czf foobar.tar lala/ lulu/`
  - `bzip2: tar cjf foobar.tar lala/ lulu/`

## ... und in den Koffer packe ich: Archivierung

- tar - Tape ARchiver
  - Archiviert Dateien
  - Stammt noch aus den Zeiten der Bandlaufwerke
  - `tar cf foobar.tar *` erzeugt Archiv
  - `tar tf foobar.tar *` zeigt Archivinhalt an
  - `tar xf foobar.tar *` extrahiert Archivinhalt
- Integration mit Kompressionsprogrammen
  - `gzip: tar czf foobar.tar lala/ lulu/`
  - `bzip2: tar cjf foobar.tar lala/ lulu/`

# ... und in den Koffer packe ich: Archivierung

- tar - Tape ARchiver
  - Archiviert Dateien
  - Stammt noch aus den Zeiten der Bandlaufwerke
  - `tar cf foobar.tar *` erzeugt Archiv
  - `tar tf foobar.tar *` zeigt Archivinhalt an
  - `tar xf foobar.tar *` extrahiert Archivinhalt
- Integration mit Kompressionsprogrammen
  - `gzip: tar czf foobar.tar lala/ lulu/`
  - `bzip2: tar cjf foobar.tar lala/ lulu/`

# ... und in den Koffer packe ich: Archivierung

- tar - Tape ARchiver
  - Archiviert Dateien
  - Stammt noch aus den Zeiten der Bandlaufwerke
  - `tar cf foobar.tar *` erzeugt Archiv
  - `tar tf foobar.tar *` zeigt Archivinhalt an
  - `tar xf foobar.tar *` extrahiert Archivinhalt
- Integration mit Kompressionsprogrammen
  - `gzip: tar czf foobar.tar lala/ lulu/`
  - `bzip2: tar cjf foobar.tar lala/ lulu/`

## ... und in den Koffer packe ich: Archivierung

- tar - Tape ARchiver
  - Archiviert Dateien
  - Stammt noch aus den Zeiten der Bandlaufwerke
  - `tar cf foobar.tar *` erzeugt Archiv
  - `tar tf foobar.tar *` zeigt Archivinhalt an
  - `tar xf foobar.tar *` extrahiert Archivinhalt
- Integration mit Kompressionsprogrammen
  - `gzip: tar czf foobar.tar lala/ lulu/`
  - `bzip2: tar cjf foobar.tar lala/ lulu/`



# Agenda

- 1 Einleitung
  - Wie sieht's aus?
  - Verzeichnisbaum
- 2 Getting started...
- 3 Editoren
- 4 Die Umgebung der Bash
- 5 Kommandos**
  - alltägliche Befehle
  - Rechte und Dateisystem**
  - Dateien Suchen
- 6 Shell für Fortgeschrittene
  - Job Control
  - Ausgabeumleitung



# Berechtigungen erkennen

## Ausgabe von `ls -l`

```
#[ makii @ whitestar : ~/tmp ]
```

```
#[0]-> ls -l
```

```
total 32
```

```
drwxr-xr-x  2 makii makii 4096 2006-11-25 21:33 css
```

```
drwxr-xr-x  2 makii makii 4096 2006-10-18 19:14 images
```

```
-rw-r--r--  1 makii makii   44 2005-12-22 17:15 index.html
```

```
-rw-r--r--  1 makii makii 2452 2006-10-18 18:47 index.php
```

```
drwxr-xr-x  2 makii makii 4096 2007-02-21 01:41 mp3
```

```
-rw-r--r--  1 makii makii 1423 2006-10-18 19:37 templateDetails.xml
```

```
-rw-r--r--  1 makii makii 4109 2006-10-18 19:10 template_thumbnail.png
```

```
#[ makii @ whitestar : ~/tmp ]
```

```
#[0]->
```



# Berechtigungen verstehen

```
drwxr-xr-x  
-rw-r--r--
```

- **d** Dateityp im Dateisystem
  - `d` Verzeichnis
  - `-` Normale Datei
  - `l` Link
- **rwX**, und das dreimal! Berechtigungen für Besitzer, Gruppe und alle anderen
  - `r` für lesen (Read)
  - `w` für schreiben (Write)
  - `x` für ausführen oder betreten (eXecute)



# Berechtigungen verstehen

```
drwxr-xr-x  
-rw-r--r--
```

- **d** Dateityp im Dateisystem
  - `d` Verzeichnis
  - `-` Normale Datei
  - `l` Link
- **rwX**, und das dreimal! Berechtigungen für Besitzer, Gruppe und alle anderen
  - `r` für lesen (Read)
  - `w` für schreiben (Write)
  - `x` für ausführen oder betreten (eXecute)



# Berechtigungen verstehen

```
drwxr-xr-x  
-rw-r--r--
```

- **d** Dateityp im Dateisystem
  - d Verzeichnis
  - - Normale Datei
  - l Link
- **rwX**, und das dreimal! Berechtigungen für Besitzer, Gruppe und alle anderen
  - r für lesen (Read)
  - w für schreiben (Write)
  - x für ausführen oder betreten (eXecute)



# Berechtigungen verstehen

```
drwxr-xr-x  
-rw-r--r--
```

- **d** Dateityp im Dateisystem
  - d Verzeichnis
  - - Normale Datei
  - l Link
- **rwX**, und das dreimal! Berechtigungen für Besitzer, Gruppe und alle anderen
  - r für lesen (Read)
  - w für schreiben (Write)
  - x für ausführen oder betreten (eXecute)



# Berechtigungen verstehen

```
drwxr-xr-x  
-rw-r--r--
```

- **d** Dateityp im Dateisystem
  - d Verzeichnis
  - - Normale Datei
  - l Link
- **rwX**, und das dreimal! Berechtigungen für Besitzer, Gruppe und alle anderen
  - r für lesen (Read)
  - w für schreiben (Write)
  - x für ausführen oder betreten (eXecute)



# Berechtigungen verstehen

```
drwxr-xr-x  
-rw-r--r--
```

- **d** Dateityp im Dateisystem
  - d Verzeichnis
  - - Normale Datei
  - l Link
- **rwX**, und das dreimal! Berechtigungen für Besitzer, Gruppe und alle anderen
  - r für lesen (Read)
  - w für schreiben (Write)
  - x für ausführen oder betreten (eXecute)



# Berechtigungen verstehen

```
drwxr-xr-x  
-rw-r--r--
```

- **d** Dateityp im Dateisystem
  - d Verzeichnis
  - - Normale Datei
  - l Link
- **rwX**, und das dreimal! Berechtigungen für Besitzer, Gruppe und alle anderen
  - r für lesen (Read)
  - w für schreiben (Write)
  - x für ausführen oder betreten (eXecute)

# Erweiterte Dateattribute

- `stat` Zeigt detailliertere Dateattribute an

```
#-[ makii @ whitestar : ~/tmp ]  
#-[0]-> stat template_thumbnail.png  
File: 'template_thumbnail.png'  
Size: 4109          Blocks: 16          IO Block: 4096   regular file  
Device: 306h/774d   Inode: 4326654      Links: 1  
Access: (0644/-rw-r--r--)  Uid: ( 1000/   makii)   Gid: ( 1000/   makii)  
Access: 2006-10-18 19:10:16.000000000 +0200  
Modify: 2006-10-18 19:10:16.000000000 +0200  
Change: 2006-11-25 21:25:27.000000000 +0100  
  
#-[ makii @ whitestar : ~/tmp ]  
#-[0]->
```

- Man beachte die alternative Schreibweise 0644 !

# Erweiterte Dateattribute

- `stat` Zeigt detailliertere Dateattribute an

```
#-[ makii @ whitestar : ~/tmp ]
#-[0]-> stat template_thumbnail.png
  File: 'template_thumbnail.png'
  Size: 4109          Blocks: 16          IO Block: 4096   regular file
Device: 306h/774d    Inode: 4326654      Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/   makii)   Gid: ( 1000/   makii)
Access: 2006-10-18 19:10:16.000000000 +0200
Modify: 2006-10-18 19:10:16.000000000 +0200
Change: 2006-11-25 21:25:27.000000000 +0100

#-[ makii @ whitestar : ~/tmp ]
#-[0]->
```

- Man beachte die alternative Schreibweise 0644 !

# Berechtigungen im Binärsystem

Dateiberechtigungen im Binärsystem anzugeben ist nicht notwendig, aber einfacher und erleichtert die Modifikationen.

	Vollzugriff	Nur Lesen	Lesen/Schreiben	Lesen/Ausführen
Textnotation	<code>rwx</code>	<code>r-</code>	<code>rw-</code>	<code>r-x</code>
<i>Wertigkeit</i>	<code>421</code>	<code>421</code>	<code>421</code>	<code>421</code>
Numerisch	<code>7</code>	<code>4</code>	<code>6</code>	<code>5</code>



# Berechtigungen verändern

- `id` zeigt an wer man gerade ist
- `su` wechselt den aktuellen Benutzer
- `stat` zeigt detailliertere Informationen zu Dateien an
- `chmod` ändert Zugriff auf Dateien
- `chgrp` ändert nur die Gruppe
- `chown` ändert Besitzer oder Gruppe
  - Dateien einem anderen Besitzer zuordnen darf nur `root`



# Berechtigungen verändern

- `id` zeigt an wer man gerade ist
- `su` wechselt den aktuellen Benutzer
- `stat` zeigt detailliertere Informationen zu Dateien an
- `chmod` ändert Zugriff auf Dateien
- `chgrp` ändert nur die Gruppe
- `chown` ändert Besitzer oder Gruppe
  - Dateien einem anderen Besitzer zuordnen darf nur `root`



# Berechtigungen verändern

- `id` zeigt an wer man gerade ist
- `su` wechselt den aktuellen Benutzer
- `stat` zeigt detailliertere Informationen zu Dateien an
- `chmod` ändert Zugriff auf Dateien
- `chgrp` ändert nur die Gruppe
- `chown` ändert Besitzer oder Gruppe
  - Dateien einem anderen Besitzer zuordnen darf nur `root`



# Berechtigungen verändern

- `id` zeigt an wer man gerade ist
- `su` wechselt den aktuellen Benutzer
- `stat` zeigt detailliertere Informationen zu Dateien an
- `chmod` ändert Zugriff auf Dateien
- `chgrp` ändert nur die Gruppe
- `chown` ändert Besitzer oder Gruppe
  - Dateien einem anderen Besitzer zuordnen darf nur `root`



# Berechtigungen verändern

- `id` zeigt an wer man gerade ist
- `su` wechselt den aktuellen Benutzer
- `stat` zeigt detailliertere Informationen zu Dateien an
- `chmod` ändert Zugriff auf Dateien
- `chgrp` ändert nur die Gruppe
- `chown` ändert Besitzer oder Gruppe
  - Dateien einem anderen Besitzer zuordnen darf nur `root`



# Berechtigungen verändern

- `id` zeigt an wer man gerade ist
- `su` wechselt den aktuellen Benutzer
- `stat` zeigt detailliertere Informationen zu Dateien an
- `chmod` ändert Zugriff auf Dateien
- `chgrp` ändert nur die Gruppe
- `chown` ändert Besitzer oder Gruppe
  - Dateien einem anderen Besitzer zuordnen darf nur `root`



# Berechtigungen verändern

- `id` zeigt an wer man gerade ist
- `su` wechselt den aktuellen Benutzer
- `stat` zeigt detailliertere Informationen zu Dateien an
- `chmod` ändert Zugriff auf Dateien
- `chgrp` ändert nur die Gruppe
- `chown` ändert Besitzer oder Gruppe
  - Dateien einem anderen Besitzer zuordnen darf nur `root`



# Agenda

- 1 Einleitung
  - Wie sieht's aus?
  - Verzeichnisbaum
- 2 Getting started...
- 3 Editoren
- 4 Die Umgebung der Bash
- 5 Kommandos**
  - alltägliche Befehle
  - Rechte und Dateisystem
  - Dateien Suchen**
- 6 Shell für Fortgeschrittene
  - Job Control
  - Ausgabeumleitung



# Wer sucht der findet! - Ausführbares

- `type` findet ausführbare Programme
  - ist ein Shell-Builtin
  - erkennt auch Aliase
- `which` findet ausführbare Programme
  - verwendet die Umgebungsvariable `$PATH`
- `whereis`
  - findet Ausführbare Programme und Manpages dazu
  - verwendet Umgebungsvariable `$PATH` und die man-Konfiguration



# Wer sucht der findet! - Ausführbares

- `type` findet ausführbare Programme
  - ist ein Shell-Builtin
  - erkennt auch Aliase
- `which` findet ausführbare Programme
  - verwendet die Umgebungsvariable `$PATH`
- `whereis`
  - findet Ausführbare Programme und Manpages dazu
  - verwendet Umgebungsvariable `$PATH` und die man-Konfiguration



# Wer sucht der findet! - Ausführbares

- `type` findet ausführbare Programme
  - ist ein Shell-Builtin
  - erkennt auch Aliase
- `which` findet ausführbare Programme
  - verwendet die Umgebungsvariable `$PATH`
- `whereis`
  - findet Ausführbare Programme und Manpages dazu
  - verwendet Umgebungsvariable `$PATH` und die man-Konfiguration



# Wer sucht der findet! - Ausführbares

- `type` findet ausführbare Programme
  - ist ein Shell-Builtin
  - erkennt auch Aliase
- `which` findet ausführbare Programme
  - verwendet die Umgebungsvariable `$PATH`
- `whereis`
  - findet Ausführbare Programme und Manpages dazu
  - verwendet Umgebungsvariable `$PATH` und die man-Konfiguration



# Wer sucht der findet! - locate

- `locate` findet Dateien über eigenen Suchindex
  - Sucht in einem Suchindex
  - Index wird von einem Job regelmäßig aktualisiert
  - `locate foobar`
  - Schneller, aber nur nach Name

# Wer sucht der findet! - locate

- `locate` findet Dateien über eigenen Suchindex
  - Sucht in einem Suchindex
  - Index wird von einem Job regelmäßig aktualisiert
  - `locate foobar`
  - Schneller, aber nur nach Name



# Wer sucht der findet! - locate

- `locate` findet Dateien über eigenen Suchindex
  - Sucht in einem Suchindex
  - Index wird von einem Job regelmäßig aktualisiert
  - `locate foobar`
  - Schneller, aber nur nach Name

# Wer sucht der findet! - locate

- `locate` findet Dateien über eigenen Suchindex
  - Sucht in einem Suchindex
  - Index wird von einem Job regelmäßig aktualisiert
  - `locate foobar`
  - Schneller, aber nur nach Name



# Wer sucht der findet! - locate

- `locate` findet Dateien über eigenen Suchindex
  - Sucht in einem Suchindex
  - Index wird von einem Job regelmäßig aktualisiert
  - `locate foobar`
  - Schneller, aber nur nach Name

# Wer sucht der findet! - find

- `find` findet Dateien in Unterverzeichnissen

- Sucht auf der Platte
- `find . -type f -name "*.txt"`
- `find . -type f -exec grep -Hni foobar`
- `find . -type d -name "bin"`
- `find / -type f -mtime +5`
- Langsamer, aber flexibler



# Wer sucht der findet! - find

- `find` findet Dateien in Unterverzeichnissen

- Sucht auf der Platte

- `find . -type f -name "*.txt"`

- `find . -type f -exec grep -Hni foobar`

- `find . -type d -name "bin"`

- `find / -type f -mtime +5`

- Langsamer, aber flexibler



# Wer sucht der findet! - find

- `find` findet Dateien in Unterverzeichnissen
  - Sucht auf der Platte
  - `find . -type f -name "*.txt"`
  - `find . -type f -exec grep -Hni foobar`
  - `find . -type d -name "bin"`
  - `find / -type f -mtime +5`
  - Langsamer, aber flexibler



# Wer sucht der findet! - find

- `find` findet Dateien in Unterverzeichnissen
  - Sucht auf der Platte
  - `find . -type f -name "*.txt"`
  - `find . -type f -exec grep -Hni foobar`
  - `find . -type d -name "bin"`
  - `find / -type f -mtime +5`
  - Langsamer, aber flexibler



# Wer sucht der findet! - find

- `find` findet Dateien in Unterverzeichnissen
  - Sucht auf der Platte
  - `find . -type f -name "*.txt"`
  - `find . -type f -exec grep -Hni foobar`
  - `find . -type d -name "bin"`
  - `find / -type f -mtime +5`
  - Langsamer, aber flexibler



# Wer sucht der findet! - find

- `find` findet Dateien in Unterverzeichnissen
  - Sucht auf der Platte
  - `find . -type f -name "*.txt"`
  - `find . -type f -exec grep -Hni foobar`
  - `find . -type d -name "bin"`
  - `find / -type f -mtime +5`
  - Langsamer, aber flexibler



# Wer sucht der findet! - find

- `find` findet Dateien in Unterverzeichnissen
  - Sucht auf der Platte
  - `find . -type f -name "*.txt"`
  - `find . -type f -exec grep -Hni foobar`
  - `find . -type d -name "bin"`
  - `find / -type f -mtime +5`
  - Langsamer, aber flexibler



# Agenda

- 1 Einleitung
  - Wie sieht's aus?
  - Verzeichnisbaum
- 2 Getting started...
- 3 Editoren
- 4 Die Umgebung der Bash
- 5 Kommandos
  - alltägliche Befehle
  - Rechte und Dateisystem
  - Dateien Suchen
- 6 Shell für Fortgeschrittene**
  - Job Control**
  - Ausgabeumleitung



# Multitasking mal anders

- In einer Shell können mehrere Programme parallel ausgeführt werden.
- Mittels Befehlen und Tastenkombinationen kann zwischen den Programmen gewechselt werden.
- Ausprobieren:
  - `jobs` listet laufende Programme
  - `STRG-z` sendet in den Hintergrund
  - `fg` holt in den Vordergrund
  - `bg` lässt Programme im Hintergrund laufen



# Multitasking mal anders

- In einer Shell können mehrere Programme parallel ausgeführt werden.
- Mittels Befehlen und Tastenkombinationen kann zwischen den Programmen gewechselt werden.
- Ausprobieren:
  - `jobs` listet laufende Programme
  - `STRG-z` sendet in den Hintergrund
  - `fg` holt in den Vordergrund
  - `bg` lässt Programme im Hintergrund laufen



# Multitasking mal anders

- In einer Shell können mehrere Programme parallel ausgeführt werden.
- Mittels Befehlen und Tastenkombinationen kann zwischen den Programmen gewechselt werden.
- Ausprobieren:
  - `jobs` listet laufende Programme
  - `STRG-z` sendet in den Hintergrund
  - `fg` holt in den Vordergrund
  - `bg` lässt Programme im Hintergrund laufen



# Multitasking mal anders

- In einer Shell können mehrere Programme parallel ausgeführt werden.
- Mittels Befehlen und Tastenkombinationen kann zwischen den Programmen gewechselt werden.
- Ausprobieren:
  - `jobs` listet laufende Programme
  - `STRG-z` sendet in den Hintergrund
  - `fg` holt in den Vordergrund
  - `bg` lässt Programme im Hintergrund laufen



# Multitasking mal anders

- In einer Shell können mehrere Programme parallel ausgeführt werden.
- Mittels Befehlen und Tastenkombinationen kann zwischen den Programmen gewechselt werden.
- Ausprobieren:
  - `jobs` listet laufende Programme
  - `STRG-z` sendet in den Hintergrund
  - `fg` holt in den Vordergrund
  - `bg` lässt Programme im Hintergrund laufen



# Multitasking mal anders

- In einer Shell können mehrere Programme parallel ausgeführt werden.
- Mittels Befehlen und Tastenkombinationen kann zwischen den Programmen gewechselt werden.
- Ausprobieren:
  - `jobs` listet laufende Programme
  - `STRG-z` sendet in den Hintergrund
  - `fg` holt in den Vordergrund
  - `bg` lässt Programme im Hintergrund laufen



# Agenda

- 1 Einleitung
  - Wie sieht's aus?
  - Verzeichnisbaum
- 2 Getting started...
- 3 Editoren
- 4 Die Umgebung der Bash
- 5 Kommandos
  - alltägliche Befehle
  - Rechte und Dateisystem
  - Dateien Suchen
- 6 **Shell für Fortgeschrittene**
  - Job Control
  - **Ausgabeumleitung**



# Ausgabe-WAS?

- Jedes Programm verfügt über folgende Ein- und Ausgabestreams:
  - `stdin` - Standardeingabe
  - `stdout` - Standardausgabe
  - `stderr` - Fehlerausgabe
- Streams können einzeln umgeleitet werden:

```
#[ makii @ whitestar : ~/tmp ]  
#-[0]-> ls > foo.txt
```

```
#[ makii @ whitestar : ~/tmp ]  
#-[0]-> cat foo.txt  
css  
foo.txt  
images  
index.html  
index.php  
mp3  
templateDetails.xml  
template_thumbnail.png
```

```
#[ makii @ whitestar : ~/tmp ]
```



# Ausgabe-WAS?

- Jedes Programm verfügt über folgende Ein- und Ausgabestreams:
  - `stdin` - Standardeingabe
  - `stdout` - Standardausgabe
  - `stderr` - Fehlerausgabe
- Streams können einzeln umgeleitet werden:

```
#[ makii @ whitestar : ~/tmp ]  
#[0]-> ls > foo.txt
```

```
#[ makii @ whitestar : ~/tmp ]  
#[0]-> cat foo.txt  
css  
foo.txt  
images  
index.html  
index.php  
mp3  
templateDetails.xml  
template_thumbnail.png
```

```
#[ makii @ whitestar : ~/tmp ]
```



# Ausgabe-WAS?

- Jedes Programm verfügt über folgende Ein- und Ausgabestreams:
  - `stdin` - Standardeingabe
  - `stdout` - Standardausgabe
  - `stderr` - Fehlerausgabe
- Streams können einzeln umgeleitet werden:

```
#[ makii @ whitestar : ~/tmp ]  
#[0]-> ls > foo.txt
```

```
#[ makii @ whitestar : ~/tmp ]  
#[0]-> cat foo.txt  
css  
foo.txt  
images  
index.html  
index.php  
mp3  
templateDetails.xml  
template_thumbnail.png
```

```
#[ makii @ whitestar : ~/tmp ]
```



# Ausgabe-WAS?

- Jedes Programm verfügt über folgende Ein- und Ausgabestreams:
  - `stdin` - Standardeingabe
  - `stdout` - Standardausgabe
  - `stderr` - Fehlerausgabe
- Streams können einzeln umgeleitet werden:

```
#-[ makii @ whitestar : ~/tmp ]  
#-[0]-> ls > foo.txt
```

```
#-[ makii @ whitestar : ~/tmp ]  
#-[0]-> cat foo.txt  
css  
foo.txt  
images  
index.html  
index.php  
mp3  
templateDetails.xml  
template_thumbnail.png
```

```
#-[ makii @ whitestar : ~/tmp ]
```



# Ausgabe getrennt umleiten

- Ausgabe in Datei umleiten:

```
find . -type f > inhalt.txt
```

- Ausgabe in Datei umleiten, Fehlermeldungen unterdrücken:

```
find . -type f 2>/dev/null 1> inhalt.txt
```

- Stream "1" ist std out
- Stream "2" ist std err



# Ausgabe getrennt umleiten

- Ausgabe in Datei umleiten:  
`find . -type f > inhalt.txt`
- Ausgabe in Datei umleiten, Fehlermeldungen unterdrücken:  
`find . -type f 2>/dev/null 1> inhalt.txt`
  - Stream "1" ist `std out`
  - Stream "2" ist `std err`



# Ausgabe getrennt umleiten

- Ausgabe in Datei umleiten:  
`find . -type f > inhalt.txt`
- Ausgabe in Datei umleiten, Fehlermeldungen unterdrücken:  
`find . -type f 2>/dev/null 1> inhalt.txt`
  - Stream "1" ist std out
  - Stream "2" ist std err



# Von A nach B nach C nach D - Piping

Mehrere Programme können so auch kombiniert werden:

- `ps auxw | grep`
- `find . -name "*.gif" | grep -v lichten`
- `who | cut -d " " -f 5 | sort -r`

# Noch Fragen?

DANN RAUS DAMIT!!



## Noch Fragen?

**DANN RAUS DAMIT!!**



# Vielen Dank für Ihre Aufmerksamkeit

## Es Danken Ihnen

- Die VHS Schwabach
- Die Linux User Schwabach
  - <http://www.lusc.de>
- .. und natürlich ich!
  - <http://www.makii.de>
  - [me@makii.de](mailto:me@makii.de) oder [themakii@gmail.com](mailto:themakii@gmail.com)