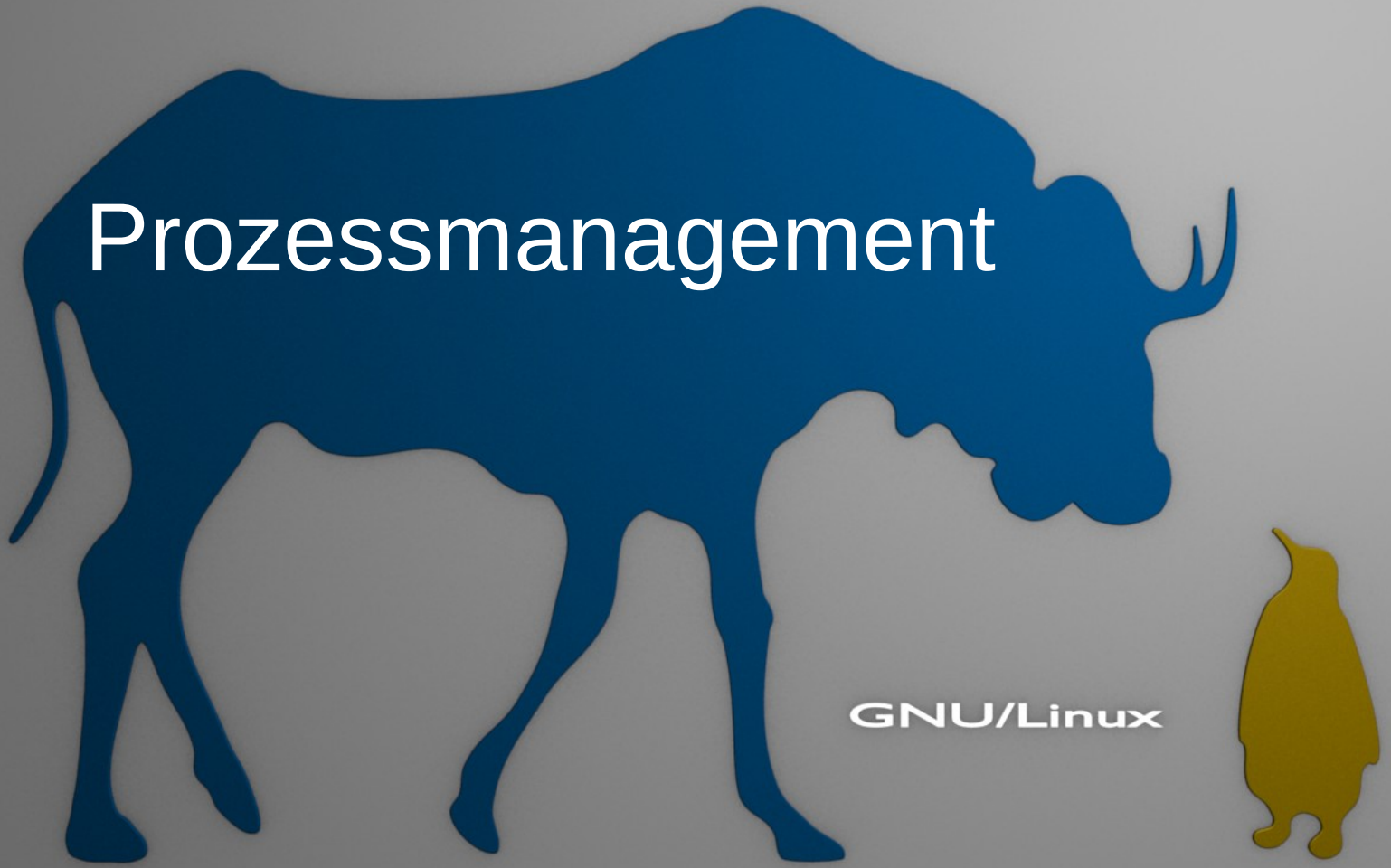
Prozessmanagement

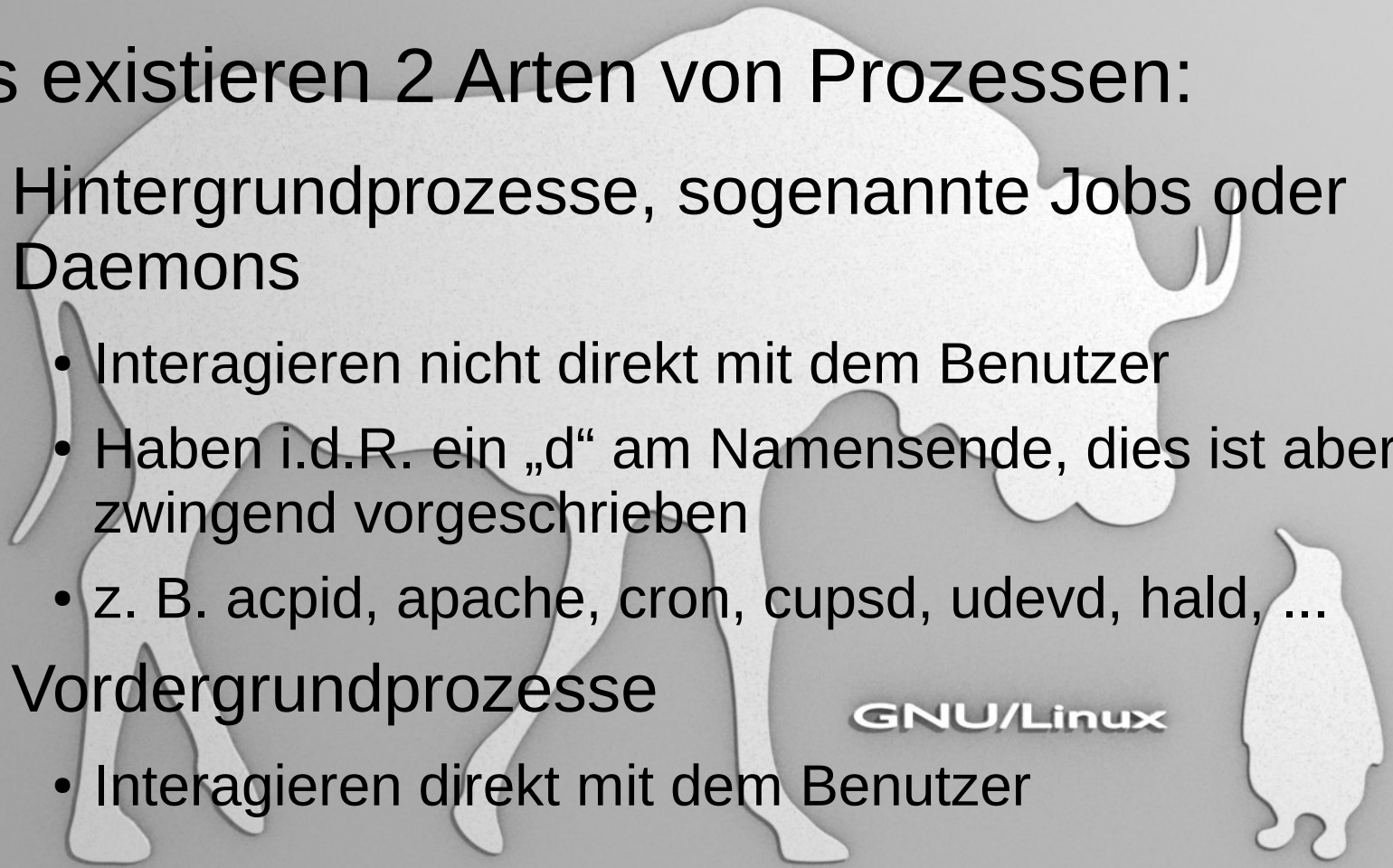


Prozessmanagement

- Jedes gestartete Programm wird durch einen Prozess abstrahiert
- Prozesse erhalten zur Laufzeit immer die Rechte des aufrufenden Benutzers
- Weiterhin wird jedem Prozess eine PID („Process ID“) zugewiesen
- Über die PID kann der Prozess später eindeutig identifiziert werden

Prozessmanagement

- Es existieren 2 Arten von Prozessen:
  - Hintergrundprozesse, sogenannte Jobs oder Daemons
    - Interagieren nicht direkt mit dem Benutzer
    - Haben i.d.R. ein „d“ am Namensende, dies ist aber nicht zwingend vorgeschrieben
    - z. B. acpid, apache, cron, cupsd, udevd, hald, ...
  - Vordergrundprozesse
    - Interagieren direkt mit dem Benutzer



Prozessmanagement

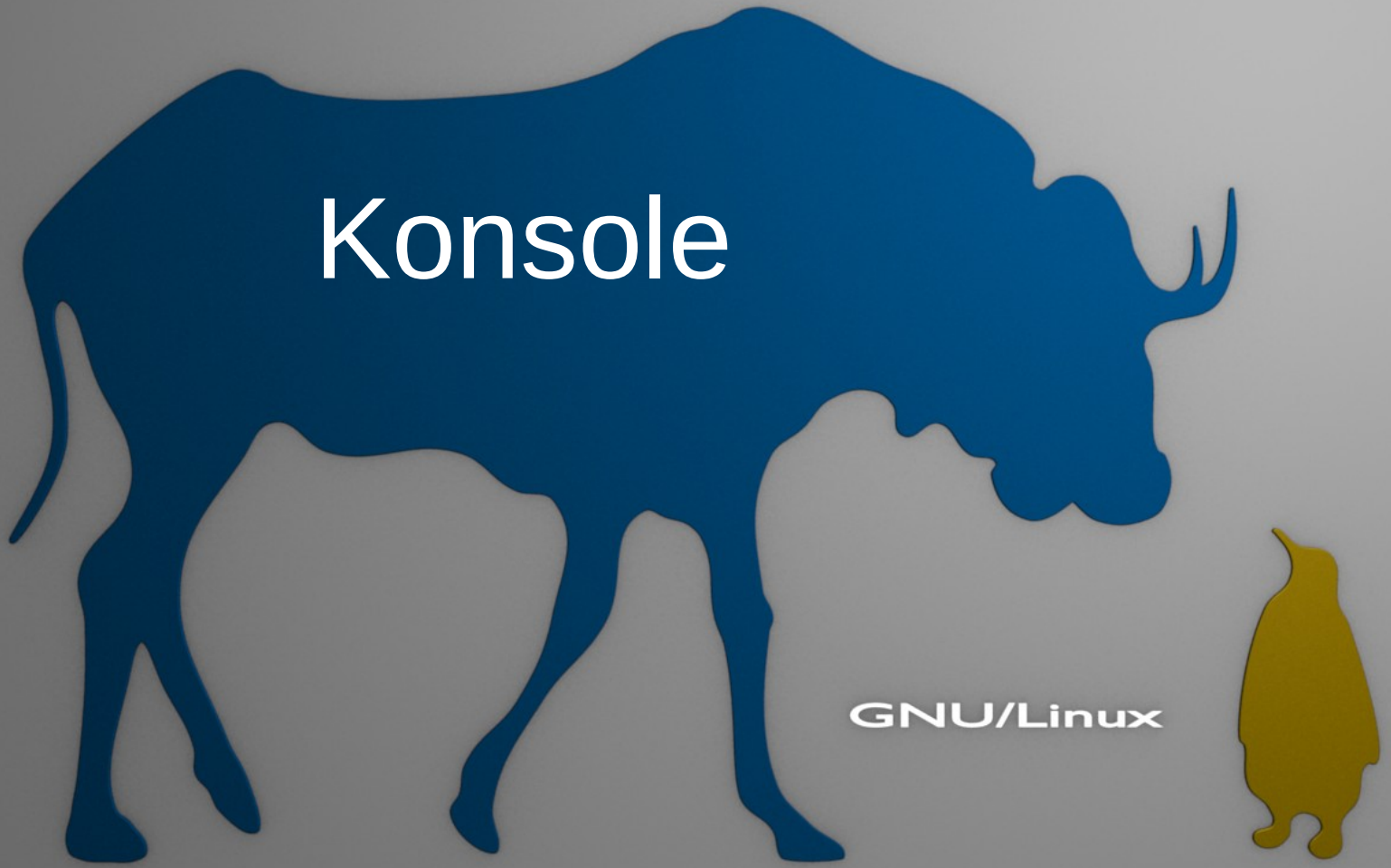
- Beim Start eines neuen Prozesses wird dieser das Kind des startenden Prozesses
- Auf diese Weise bildet sich ein Prozessbaum, an dessen Spitze der init-Prozess steht
- Wird ein Elternprozess beendet, so reist er alle Kinderprozesse mit in den „Tod“
- Übersicht über alle laufenden Prozesse liefern die Befehle „top“, „ps aux“ und „pidof programm name“ oder das grafische Tool „gnome-system-monitor“

Prozessmanagement

- Die Kommunikation mit Prozessen erfolgt über Signale, u.a.:
  - [SIG]KILL oder Nummer 9
    - Der Prozess wird sofort durch das System beendet
  - [SIG]TERM oder Nummer 15
    - Der Prozess soll sich selbst beenden
- Signale werden mit dem Befehl „kill“ versendet:

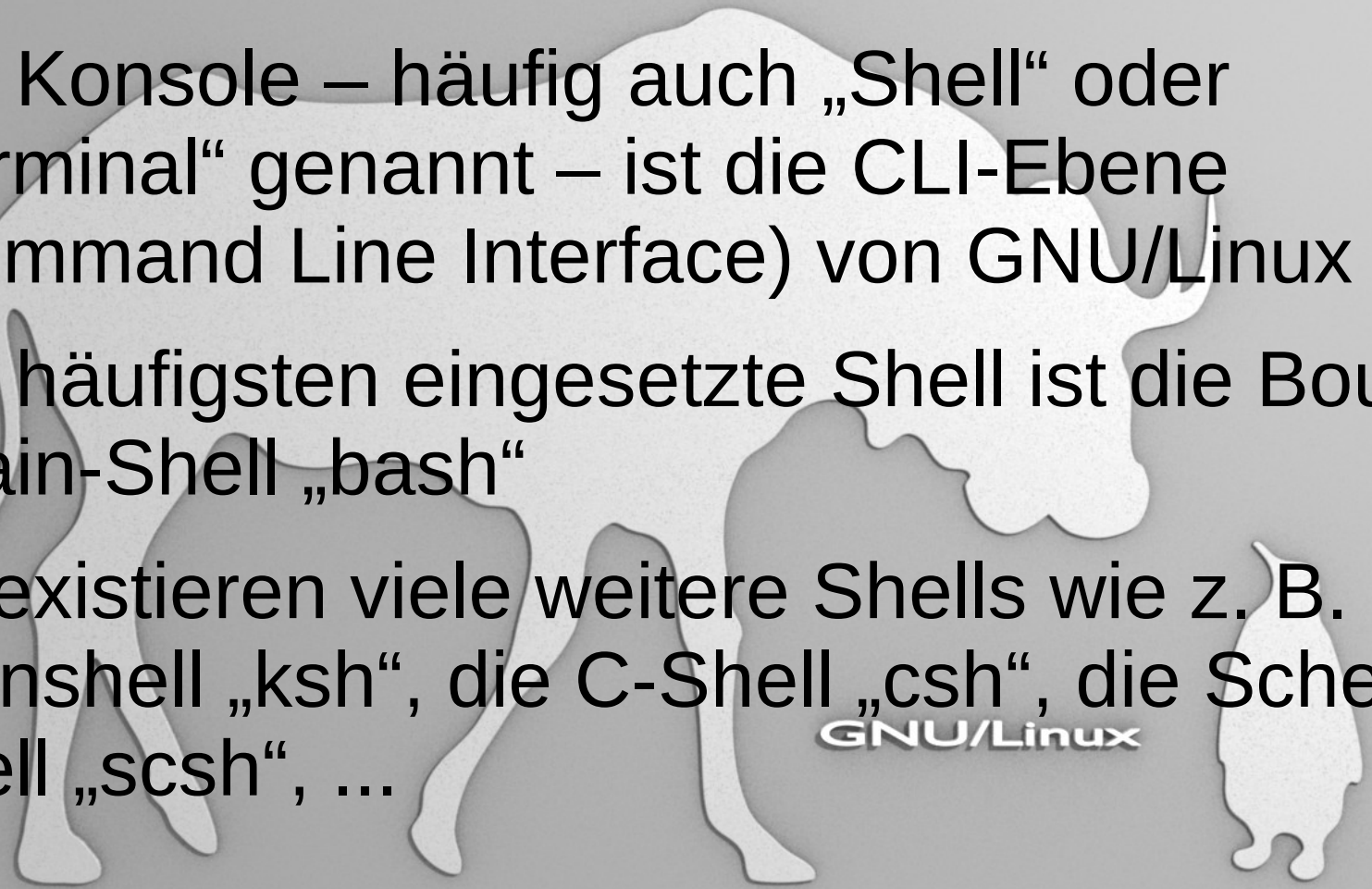
  kill -s TERM pid oder kill -15 pid
- Alternativ: killall programmname
  - Sucht in den laufenden Prozessen nach „Programmname“ und sendet an alle Prozesse mit diesem Namen das Signal TERM, wenn nicht anders angegeben

Konsole



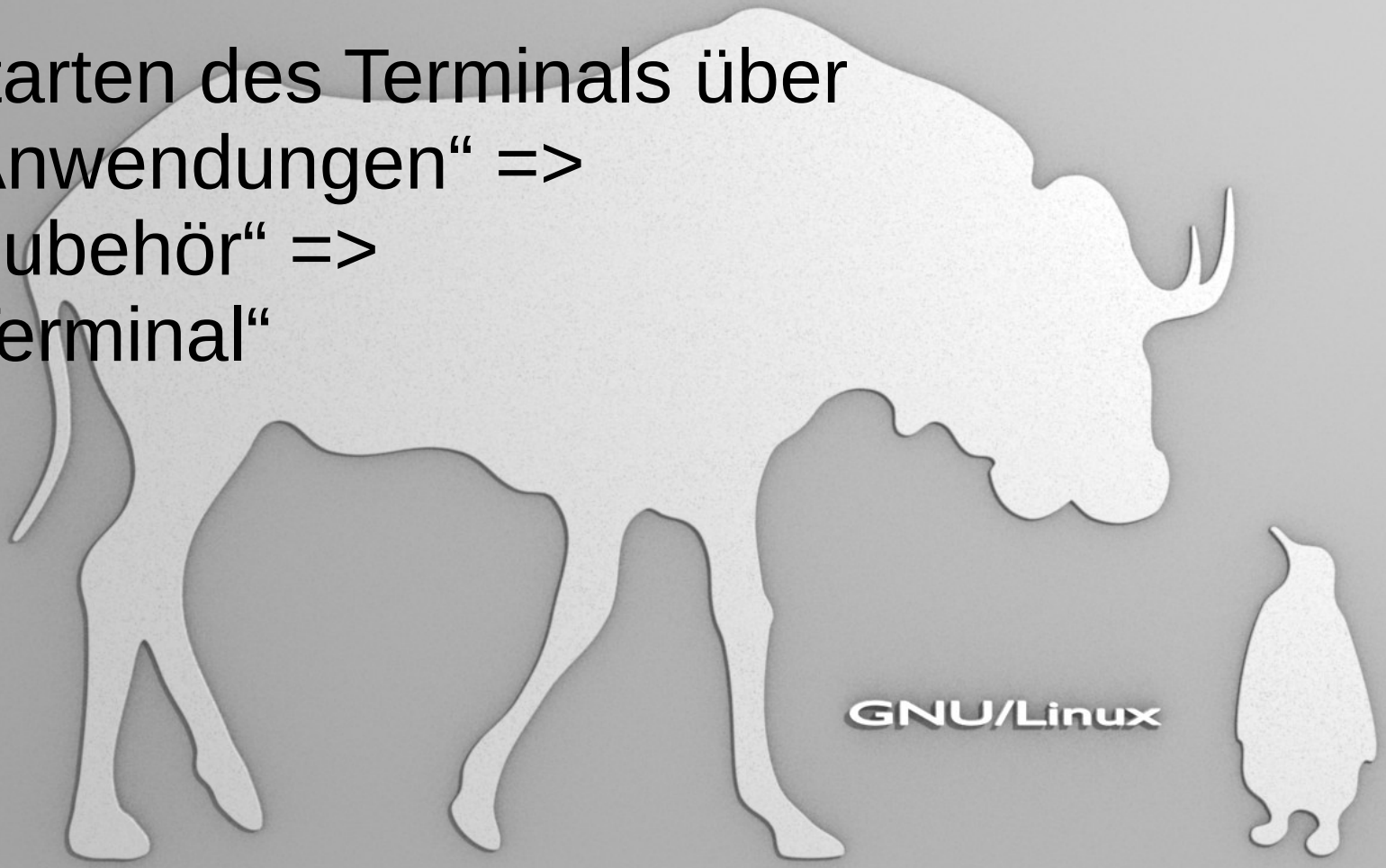
Konsole

- Die Konsole – häufig auch „Shell“ oder „Terminal“ genannt – ist die CLI-Ebene (Command Line Interface) von GNU/Linux
- Am häufigsten eingesetzte Shell ist die Bourne-Again-Shell „bash“
- Es existieren viele weitere Shells wie z. B. die Kornshell „ksh“, die C-Shell „csh“, die Scheme-Shell „scsh“, ...



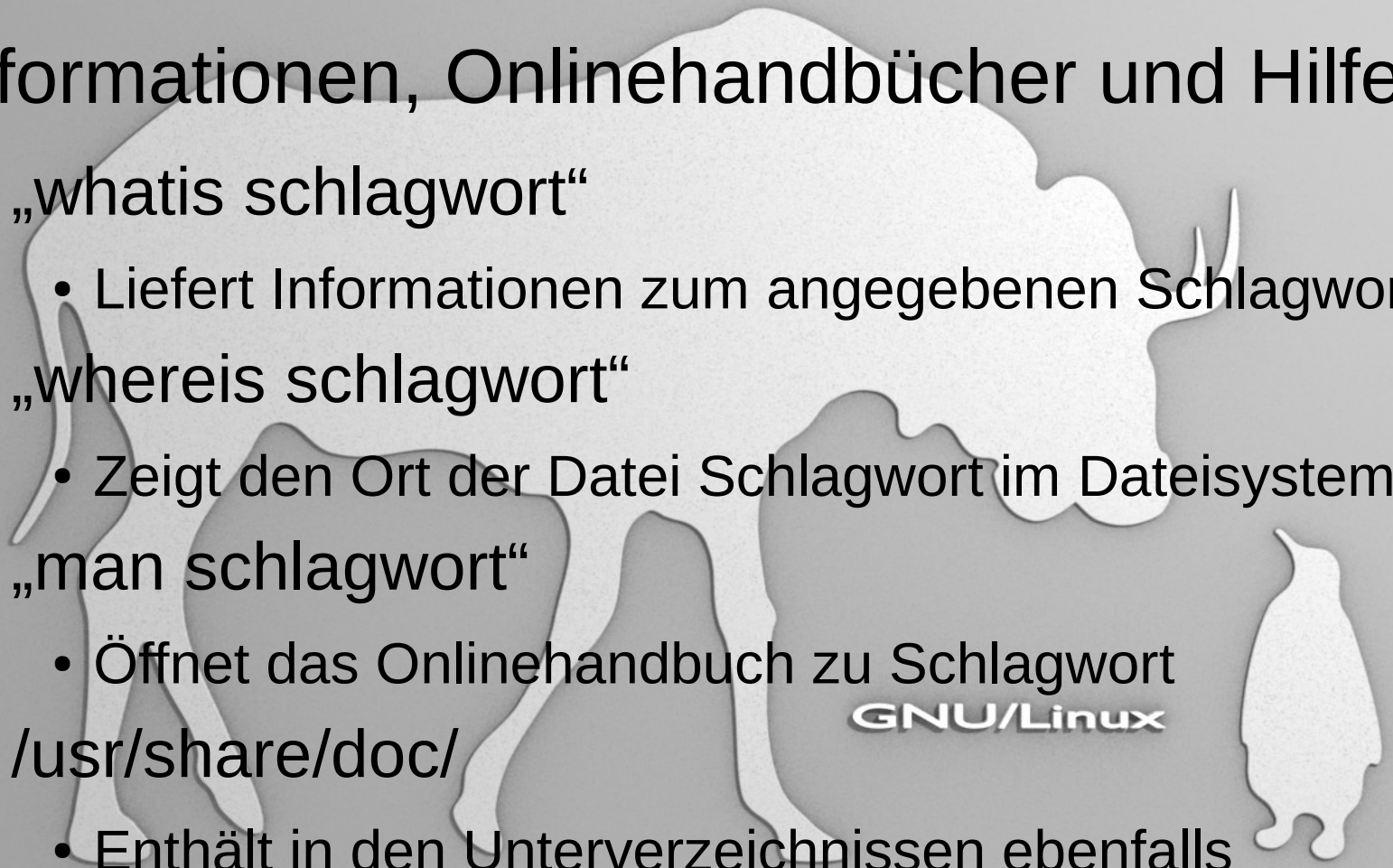
Linux Tutorium

- Starten des Terminals über „Anwendungen“ => „Zubehör“ => „Terminal“



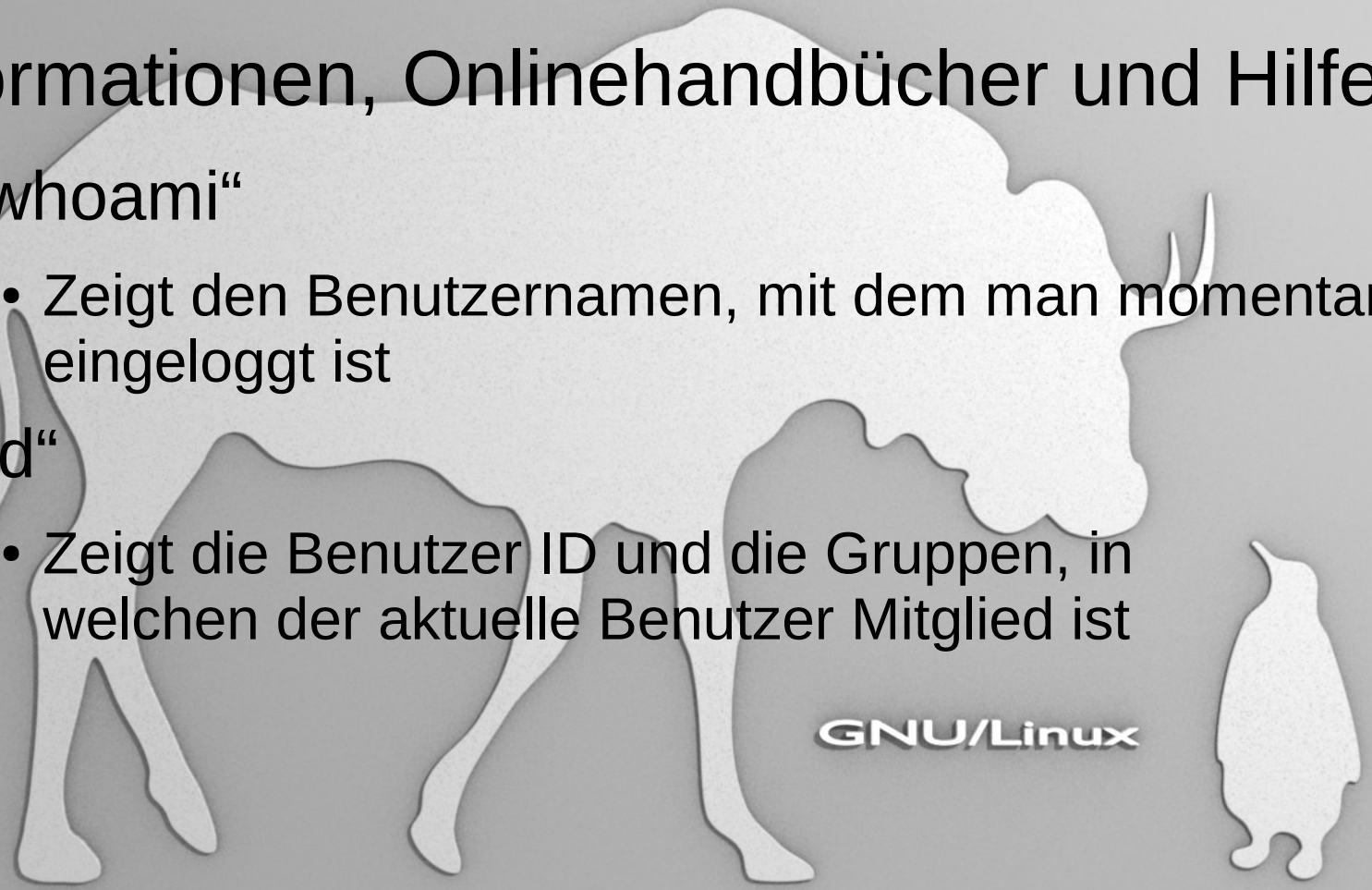
Konsole

- Informationen, Onlinehandbücher und Hilfe (1)
  - „whatis schlagwort“
    - Liefert Informationen zum angegebenen Schlagwort
  - „whereis schlagwort“
    - Zeigt den Ort der Datei Schlagwort im Dateisystem an
  - „man schlagwort“
    - Öffnet das Onlinehandbuch zu Schlagwort
  - /usr/share/doc/
    - Enthält in den Unterverzeichnissen ebenfalls Dokumentation und Beispielkonfigurationsdateien



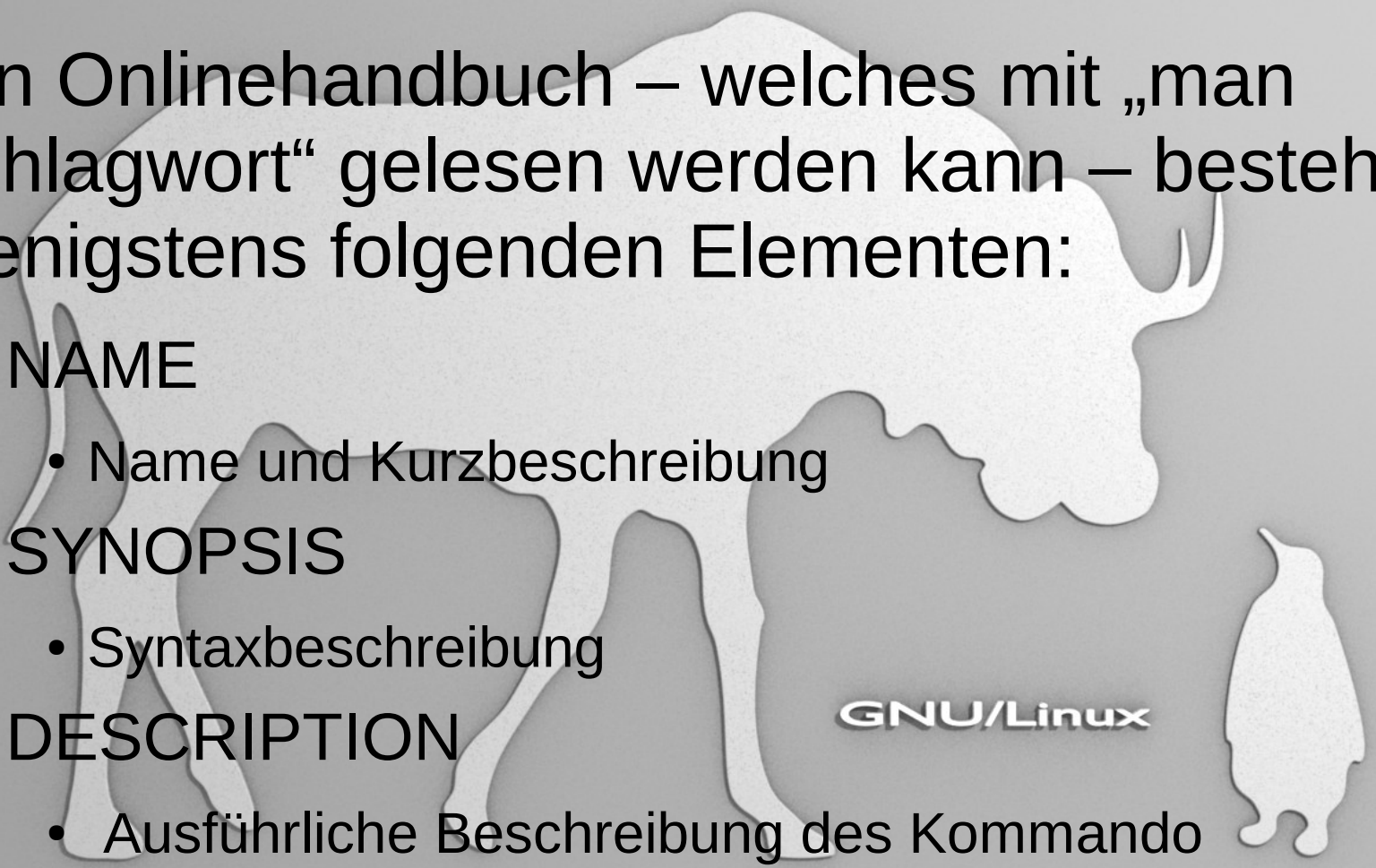
Konsole

- Informationen, Onlinehandbücher und Hilfe (2)
  - „whoami“
    - Zeigt den Benutzernamen, mit dem man momentan eingeloggt ist
  - „id“
    - Zeigt die Benutzer ID und die Gruppen, in welchen der aktuelle Benutzer Mitglied ist



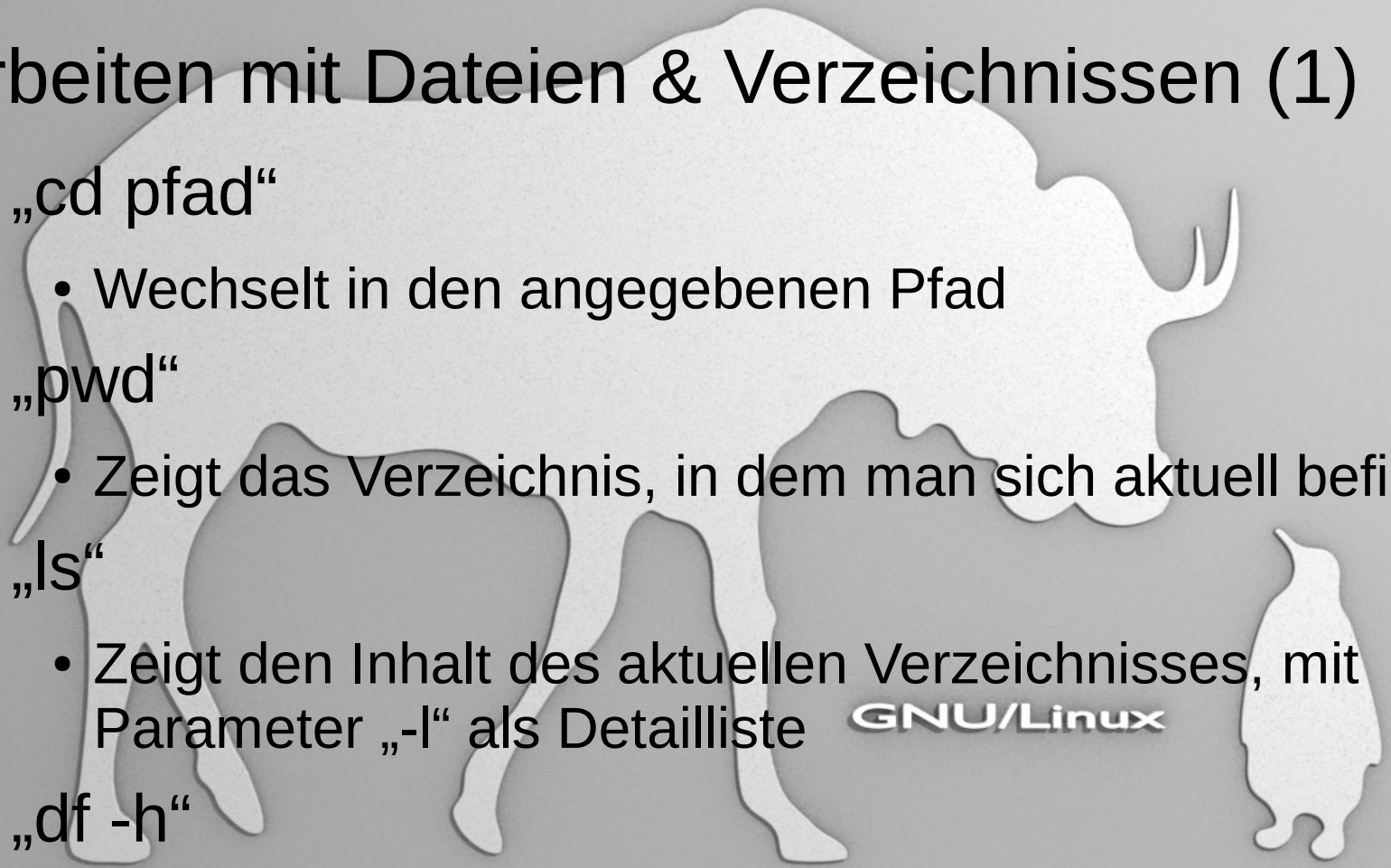
Konsole

- Ein Onlinehandbuch – welches mit „man schlagwort“ gelesen werden kann – besteht aus wenigstens folgenden Elementen:
  - NAME
    - Name und Kurzbeschreibung
  - SYNOPSIS
    - Syntaxbeschreibung
  - DESCRIPTION
    - Ausführliche Beschreibung des Kommando



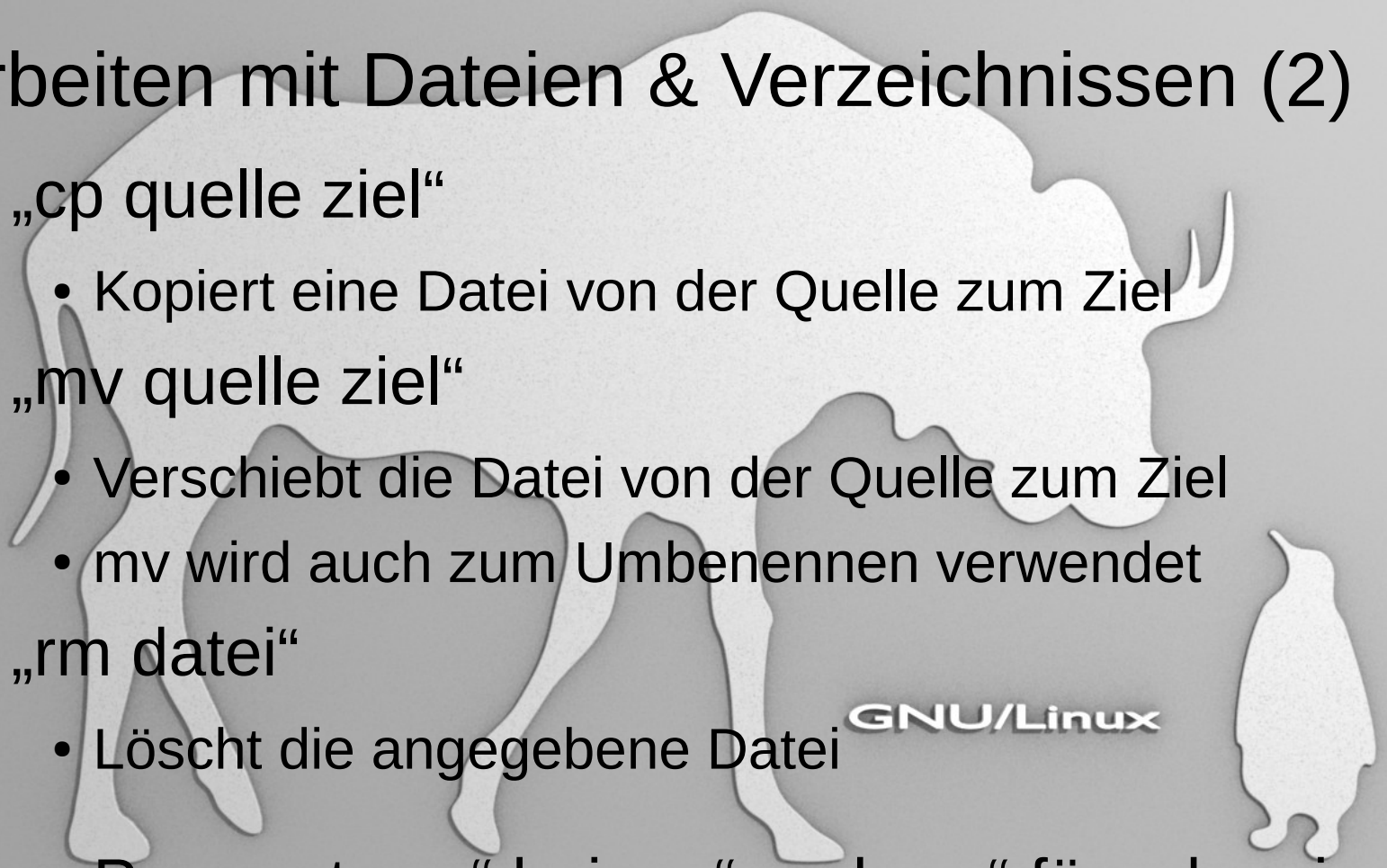
Konsole

- Arbeiten mit Dateien & Verzeichnissen (1)
  - „cd pfad“
    - Wechselt in den angegebenen Pfad
  - „pwd“
    - Zeigt das Verzeichnis, in dem man sich aktuell befindet
  - „ls“
    - Zeigt den Inhalt des aktuellen Verzeichnisses, mit Parameter „-l“ als Detailliste
  - „df -h“
    - Zeigt freien Speicherplatz auf Datenträger an
  - „clear“ oder Strg+l
    - Löscht den Bildschirminhalt



Konsole

- Arbeiten mit Dateien & Verzeichnissen (2)
  - „cp quelle ziel“
    - Kopiert eine Datei von der Quelle zum Ziel
  - „mv quelle ziel“
    - Verschiebt die Datei von der Quelle zum Ziel
    - mv wird auch zum Umbenennen verwendet
  - „rm datei“
    - Löscht die angegebene Datei



Parameter „r“ bei „cp“ und „rm“ für rekursiv

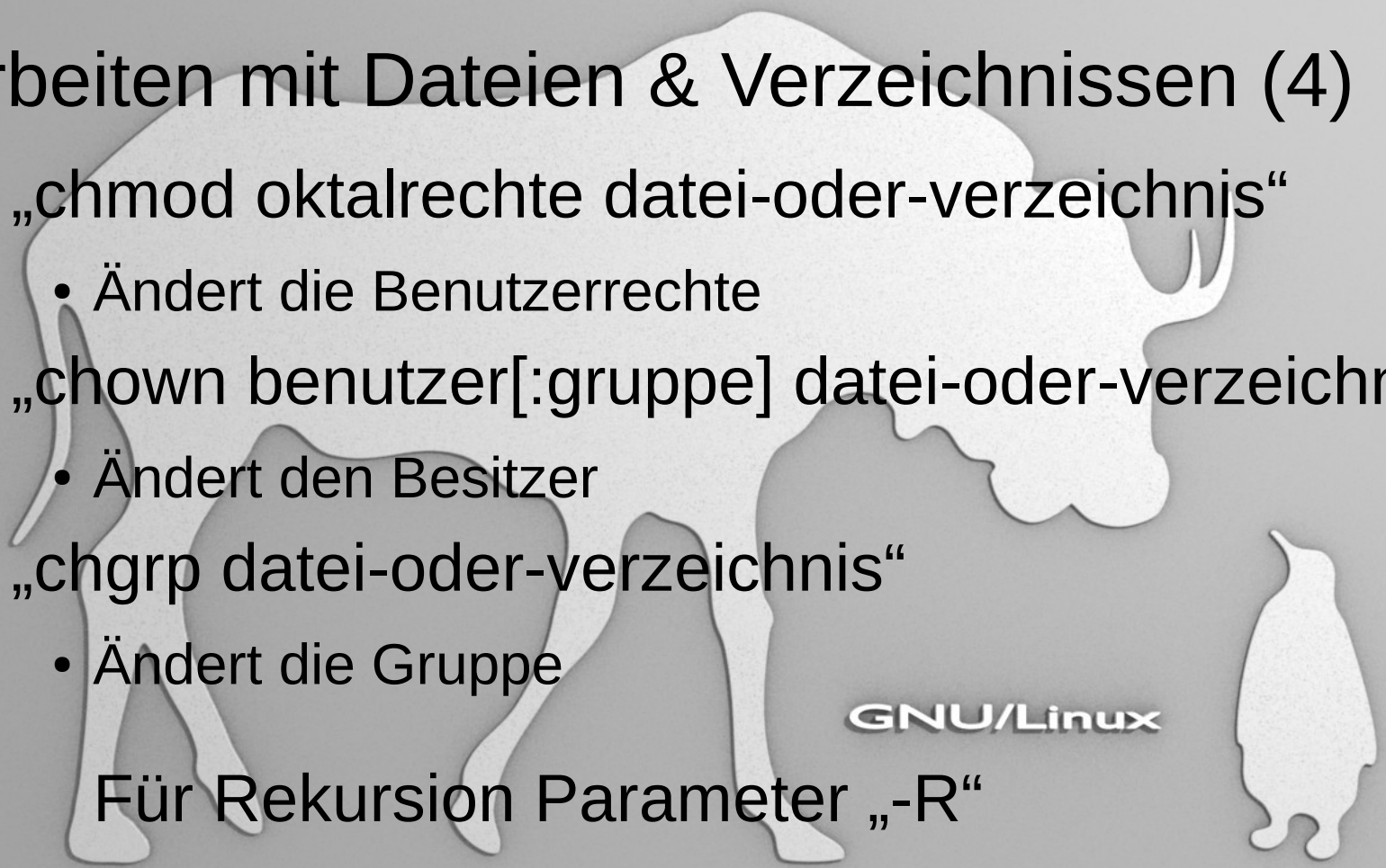
Konsole

- Arbeiten mit Dateien & Verzeichnissen (3)
  - „touch datei“ bzw. „touch verzeichnis“
    - Ändert den Zeitstempel für die Datei bzw. das angegebene Verzeichnis
    - Ist die Datei nicht vorhanden, so wird sie mit einer Größe von 0 Byte angelegt. Touch wird vorrangig zum Anlegen neuer Dateien verwendet.
  - „mkdir verzeichnis“
    - Legt ein neues Verzeichnis mit dem angegebenen Namen an
  - „ln -s ziel verknüpfungsname“
    - Erzeugt eine symbolische Verknüpfung

Konsole

- Arbeiten mit Dateien & Verzeichnissen (4)
  - „chmod oktalrechte datei-oder-verzeichnis“
    - Ändert die Benutzerrechte
  - „chown benutzer[:gruppe] datei-oder-verzeichnis“
    - Ändert den Besitzer
  - „chgrp datei-oder-verzeichnis“
    - Ändert die Gruppe

    Für Rekursion Parameter „-R“

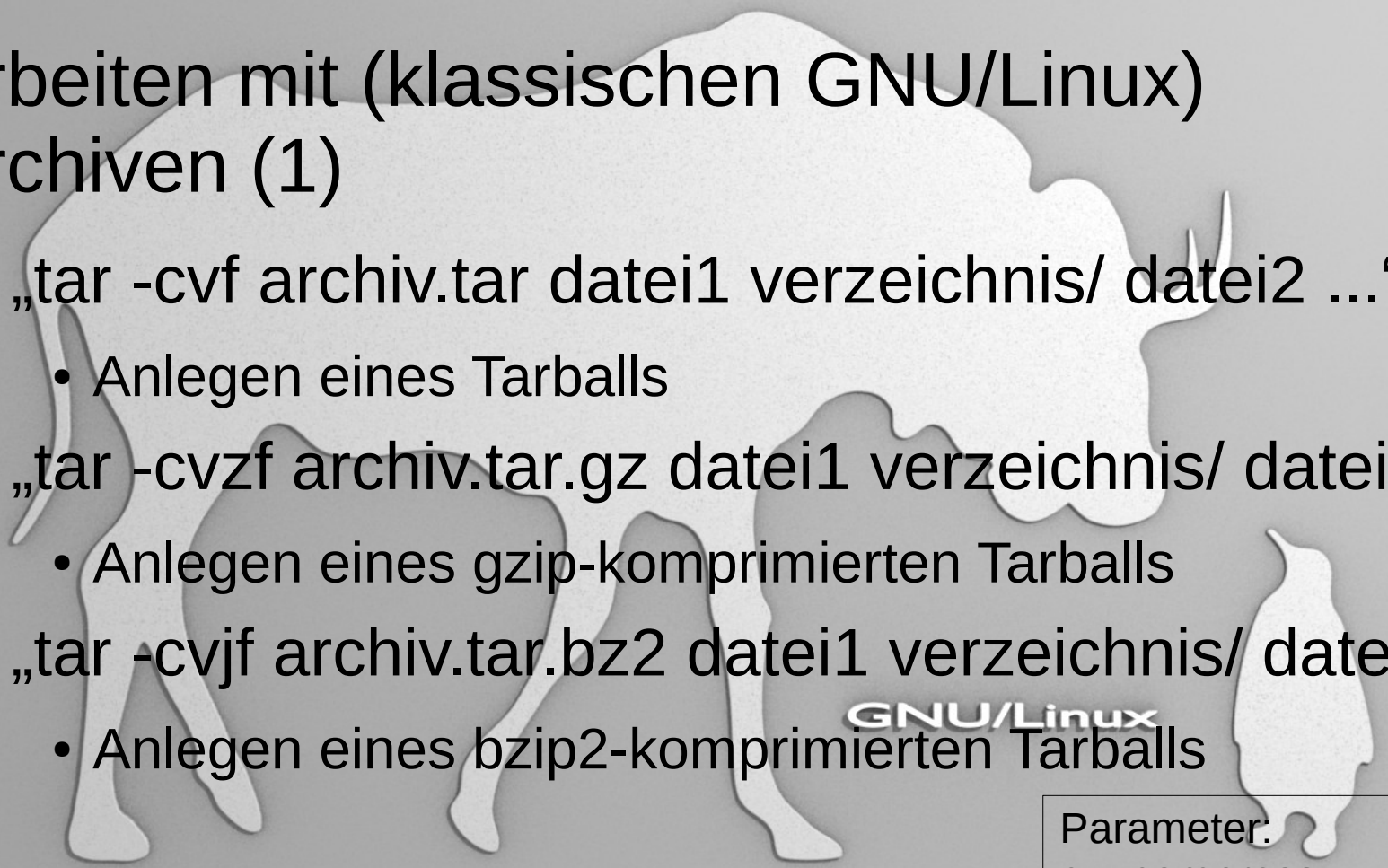


Konsole

- Suche nach und in Dateien
  - „find pfad -name "begriff"“
    - Durchsucht den Verzeichnisbaum beginnend beim angegebenen Pfad nach Dateien und Verzeichnissen, die „begriff“ im Namen enthalten und zeigt diese an. Der zusätzliche Parameter „-ls“ gibt eine Detailliste aus.
  - „grep -slr "begriff" pfad“
    - Durchsucht den Verzeichnisbaum beginnend beim angegebenen Pfad nach Dateien und Verzeichnissen, die „begriff“ als Inhalt enthalten und zeigt diese an

Konsole

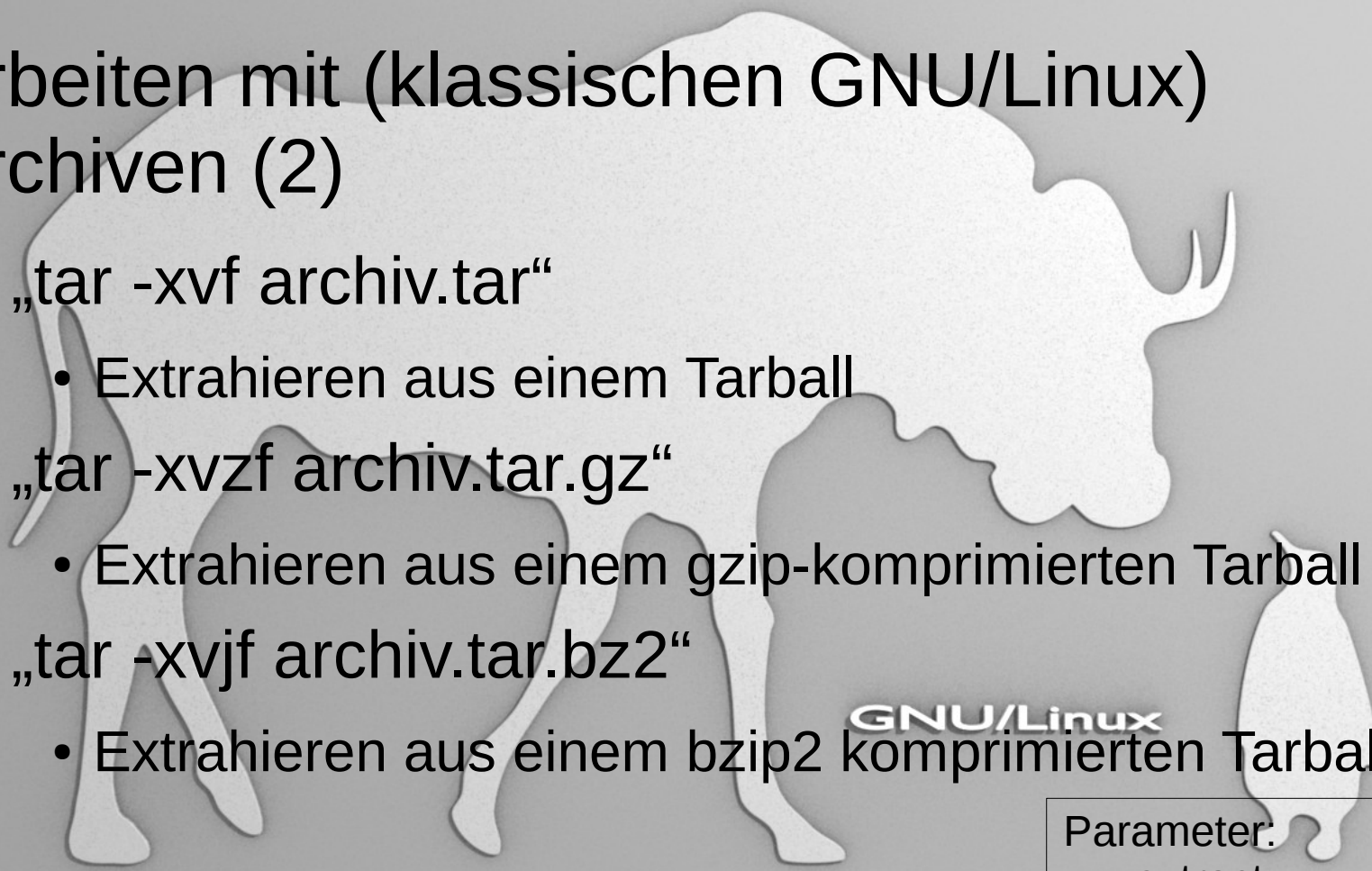
- Arbeiten mit (klassischen GNU/Linux) Archiven (1)
  - „tar -cvf archiv.tar datei1 verzeichnis/ datei2 ...“
    - Anlegen eines Tarballs
  - „tar -cvzf archiv.tar.gz datei1 verzeichnis/ datei2 ...“
    - Anlegen eines gzip-komprimierten Tarballs
  - „tar -cvjf archiv.tar.bz2 datei1 verzeichnis/ datei2 ...“
    - Anlegen eines bzip2-komprimierten Tarballs



Parameter:
c = compress
v = verbose
z = gnu zip compression
j = bzip2 compression
f = following file

Konsole

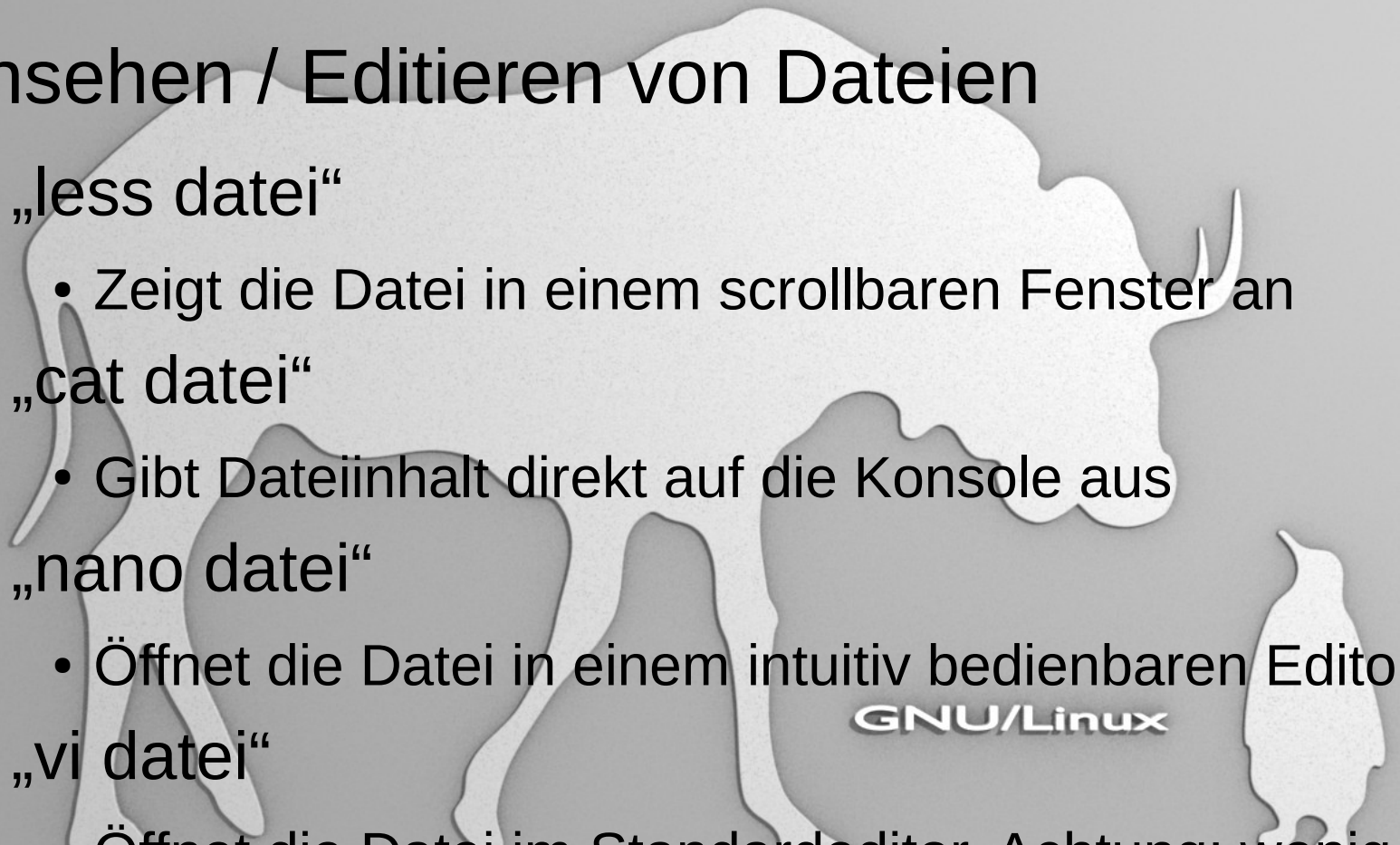
- Arbeiten mit (klassischen GNU/Linux) Archiven (2)
  - „tar -xvf archiv.tar“
    - Extrahieren aus einem Tarball
  - „tar -xvzf archiv.tar.gz“
    - Extrahieren aus einem gzip-komprimierten Tarball
  - „tar -xvjf archiv.tar.bz2“
    - Extrahieren aus einem bzip2 komprimierten Tarball



Parameter:
x = extract
v = verbose
z = gnu zip compression
j = bzip2 compression
f = following file

Konsole

- Ansehen / Editieren von Dateien
  - „less datei“
    - Zeigt die Datei in einem scrollbaren Fenster an
  - „cat datei“
    - Gibt Dateiinhalt direkt auf die Konsole aus
  - „nano datei“
    - Öffnet die Datei in einem intuitiv bedienbaren Editor
  - „vi datei“
    - Öffnet die Datei im Standardeditor. Achtung: wenig intuitiv!



Konsole

- Benutzermanagement
  - „adduser benutzername“
    - Neuen Benutzer anlegen
  - „deluser benutzername“
    - Benutzer löschen
  - „su benutzername“
    - Nach Passworteingaben zum angegebenen Benutzer werden
  - „su“
    - ohne Angabe des Benutzernamens wird zum Superuser (= root) gewechselt

Konsole

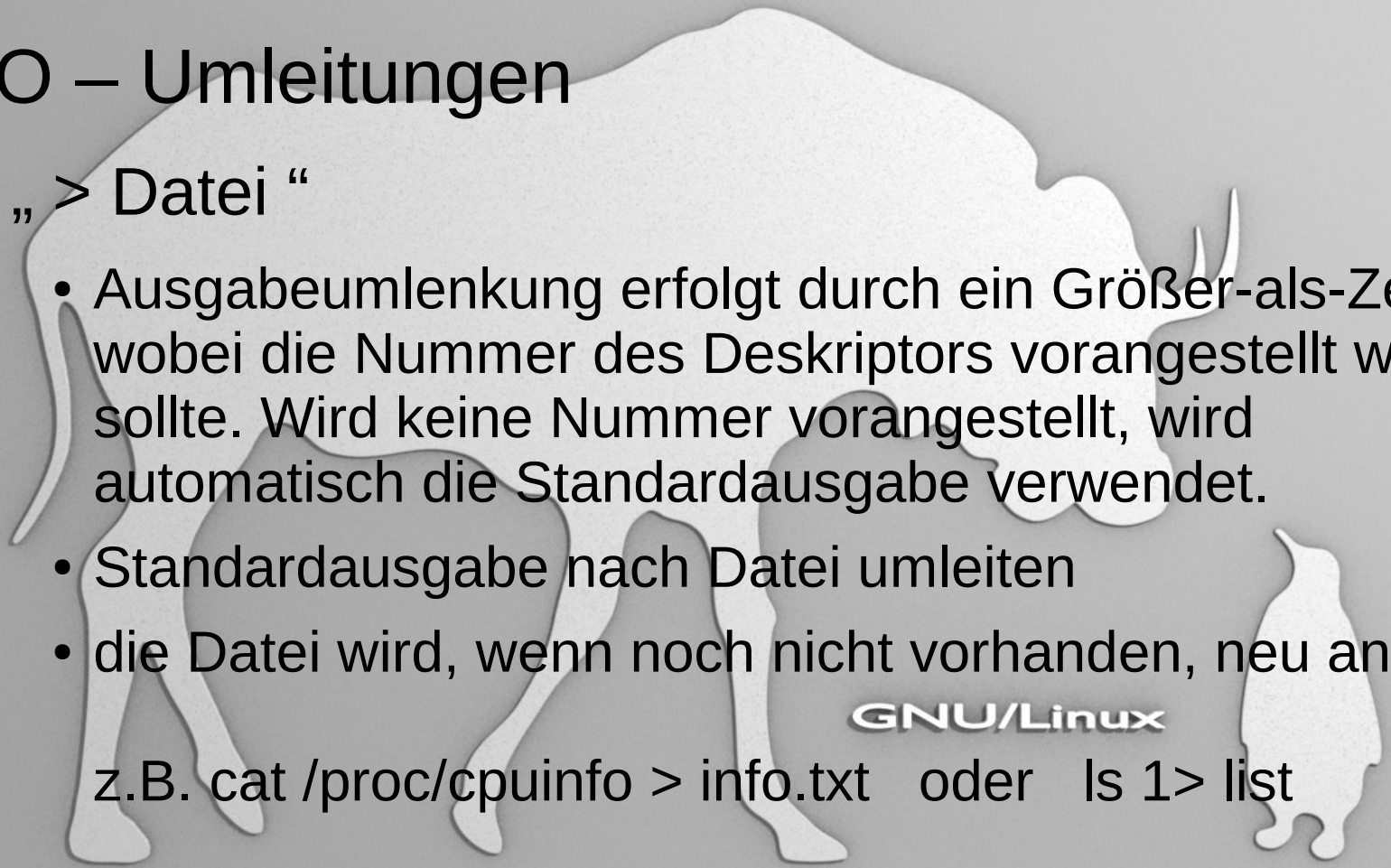
- Dateisystemmanagement
  - „mount“
    - Dient zum Einhängen / Einbinden von Dateisystemen
  - „umount“
    - Hängt eingebundene Dateisysteme wieder aus

z.B. mount /dev/sda1 /media/usbstick
umount /media/usbstick

Konsole

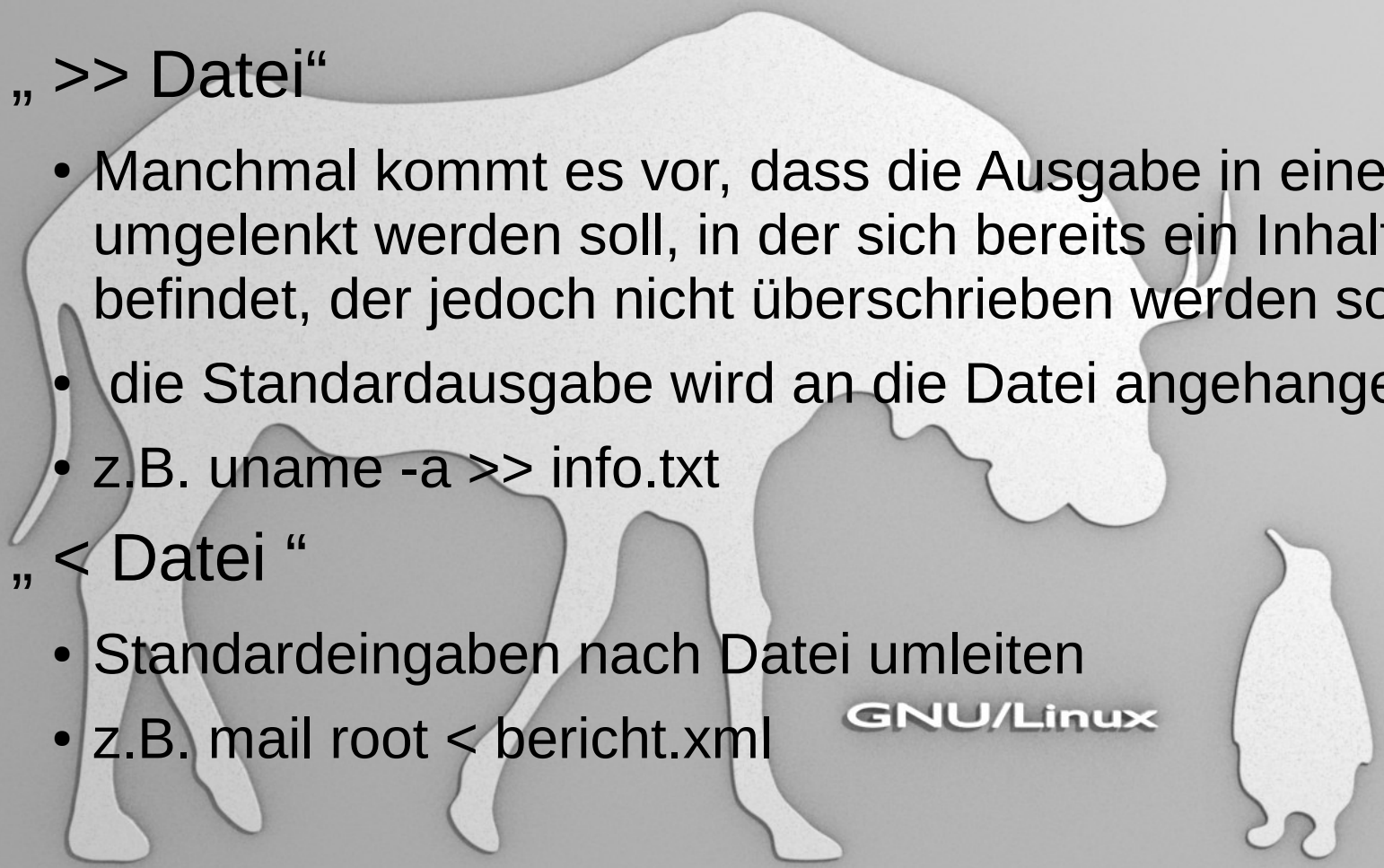
- I/O – Umleitungen
  - „ > Datei “
    - Ausgabeumlenkung erfolgt durch ein Größer-als-Zeichen, wobei die Nummer des Deskriptors vorangestellt werden sollte. Wird keine Nummer vorangestellt, wird automatisch die Standardausgabe verwendet.
    - Standardausgabe nach Datei umleiten
    - die Datei wird, wenn noch nicht vorhanden, neu angelegt

    z.B. cat /proc/cpuinfo > info.txt oder ls 1> list



Konsole

- „ >> Datei“
  - Manchmal kommt es vor, dass die Ausgabe in eine Datei umgelenkt werden soll, in der sich bereits ein Inhalt befindet, der jedoch nicht überschrieben werden soll
  - die Standardausgabe wird an die Datei angehangen
  - z.B. uname -a >> info.txt
- „ < Datei “
  - Standardeingaben nach Datei umleiten
  - z.B. mail root < bericht.xml



Konsole

- Pipes
  - es ist möglich die Standardausgabe eines Komman dos direkt einem zweiten Kommando als Standard-eingabe zu übergeben
  - im Gegensatz zur Ausgabeumlenkung wird die Aus gabe eines Programms nicht in eine Datei geschrie ben, sondern direkt an ein Programm weitergeleitet
  - Diese Weiterleitung erfolgt mit einem Pipe-Operator (|).
  - Dieser Strich wird zwischen die beiden Programme gesetzt:
    „Befehl1 | Befehl2“

Konsole

- z.B. „apt-cache show coreutils | grep Size“

```
root@ubuntu:~# apt-cache show coreutils | grep Size
Installed-Size: 11576
Size: 2319994
```

  - **Grep sucht in Kombination mit Pipe die Zeile, in der der angegebene Begriff vorkommt**

- Die Bash-History
  - sie speichert die zuletzt eingegebenen Befehle
  - man kann diese Befehle abrufen und braucht sie nicht erneut einzugeben, um sie wieder aufzurufen
  - die Bash bietet die Möglichkeit, diese Befehle zu editieren

Konsole

- eine Übersicht über die in der History enthaltenen Befehle liefert der Befehl „history“
- Cursor-Tasten erleichtern den Umgang mit der History
  - durch die Cursor-Taste »Nach oben« kann der zuletzt eingegebenen Befehl in der Kommandozeile erneut aufgerufen werden, so ist das durchlaufen der History an den Anang möglich
  - invers dazu ist dies auch mit der »Nach unten« Taste möglich, wenn man sich in der History befindet, um den gewünschten Befehl zu finden
- auch eine Suche in der History ist relativ einfach möglich

Konsole

- [Strg- r] = Rückwärtiges Suchen im Historybuffer
- [Strg-s] = Vorwärts im Historybuffer suchen

- Kommandos aneinander reihen
  - unter Linux ist es generell möglich, mehr als nur ein Kommando pro Befehl auszuführen
  - ein Befehl endet erst, wenn er mit der Eingabetaste an die Shell geschickt wird
  - ein Kommando kann jedoch durch ein Semikolon (;) von einem anderen getrennt werden
  - Die Kommandos laufen immer hintereinander ab (also auch wenn einzelne Befehle fehlschlagen)
    - z.B. „ls Verzeichnis ; uname ; find / -name Datei“

Konsole

- „dmesg“
- Anzeigen der Kontrollmeldungen des Systems aus dem Ringpuffer des Kernels
  - in diesem Puffer sind alle Kernelmeldungen seit
  - dem letzten Hochfahren (wenn der Puffer voll war nur die neusten)

```
linux@debian:~$ dmesg
[    0.000000] Initializing cgroup subsys cpu
[    0.000000] Linux version 2.6.26-1-486 (Debian 2.6.26-12) (waldi@debian.org)
(gcc version 4.1.3 20080704 (prerelease) (Debian 4.1.2-24)) #1 Mon Dec 15
17:32:01 UTC 2008
[...]
```

Konsole

- Tab-Vervollständigung
  - Erleichtet die Arbeit mit der Konsole
  - Funktionsweise:
    - den Befehl/Datei/Verzeichnis/Pfad beginnen einzugeben
    - drückt man nach ein paar Zeichen <TAB>, versucht die Shell automatisch die gewünschte Eingabe aufzufüllen
    - 3 mögliche Situationen treten nun ein:
      - 1. es kann keine Vervollständigung durchgeführt werden, der PC -Speaker gibt einen Signalton aus
      - 2. Die Vervollständigung funktioniert aufgrund der Eindeutigkeit der bisherigen Eingabe durch den User
      - 3. es liegt keine Eindeutigkeit vor, es passiert nichts. Wird <TAB> nun doppelt betätigt, werden alle in Frage kommenden Möglichkeiten aufgelistet

Linux Tutorium

- Shell-Aliase
  - erlauben es, einen neuen Befehlsnamen zu vergeben, der stellvertretend für ein Kommando oder eine Kommandoreihe steht
  - ein Alias wird über das gleichnamige Kommando „alias“ erstellt und modifiziert
  - einen Alias erstellt man, indem man den neuen Befehlsnamen sowie den Befehl, der hinter diesem verborgen sein soll, dem alias-Kommando übergibt
    - z.B. alias dir="ls -lh“

Linux Tutorium

- möchte man einen erstellten Alias löschen, ist dies einfach durch Eingabe des Kommando „unalias“ möglich, welchen man den entsprechenden Namen des Alias übergibt welcher entfernt werden soll
  - z.B. unalias dir
- eine Liste der bereits vorhandenen Alias-Einträge kann ebenfalls durch den Befehl „alias“ (ohne weitere Parameter) ausgeben werden
- Um den Alias dauerhaft zu speichen, muss er (z.B.) in die ~/.bashrc eingetragen werden