

# Vorlesung Linux-Praktikum

## 7. if-Verzweigungen und Variablen

Dirk Frettlöh

Folien nach Carsten Gnörlich

Technische Fakultät  
Universität Bielefeld

# Willkommen zur achten Vorlesung

Was gab es beim vorvorletzten Mal?

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

- ▶ Dateiverwaltung (find)
- ▶ Aliase
- ▶ Umgebungsvariablen (PATH, USER,...)
- ▶ Shellskripte

# Willkommen zur sechsten Vorlesung

Was machen wir heute?

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

## Shellskripte

Wiederholung

## Verzweigungen

if

test

grep

## Variablen

case

# Shell-Skripte

Wiederholung: Shellskript

## Prinzipieller Aufbau eines Shell-Skriptes

- ▶ Textdatei mit folgendem Inhalt:

```
#!/bin/bash
```

Shell zum Ausführen des Skriptes

```
echo Hallo  
echo ich bin ein  
echo Shellskript
```

Aufrufe, wie Ihr sie auch  
direkt eintippen würdet

# Shell-Skripte

Wiederholung: Parameterübergabe

Beispiel zur Übergabe von Parametern:

```
#!/bin/bash
```

```
echo "Erstes : $1"
```

```
echo "Zweites: $2"
```

```
echo "Drittes: $3"
```

```
echo "Anzahl : $#"
```

```
echo "Alle   : $*"
```

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

# Shell-Skripte

Wiederholung: Datei mit Überschrift sortieren

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

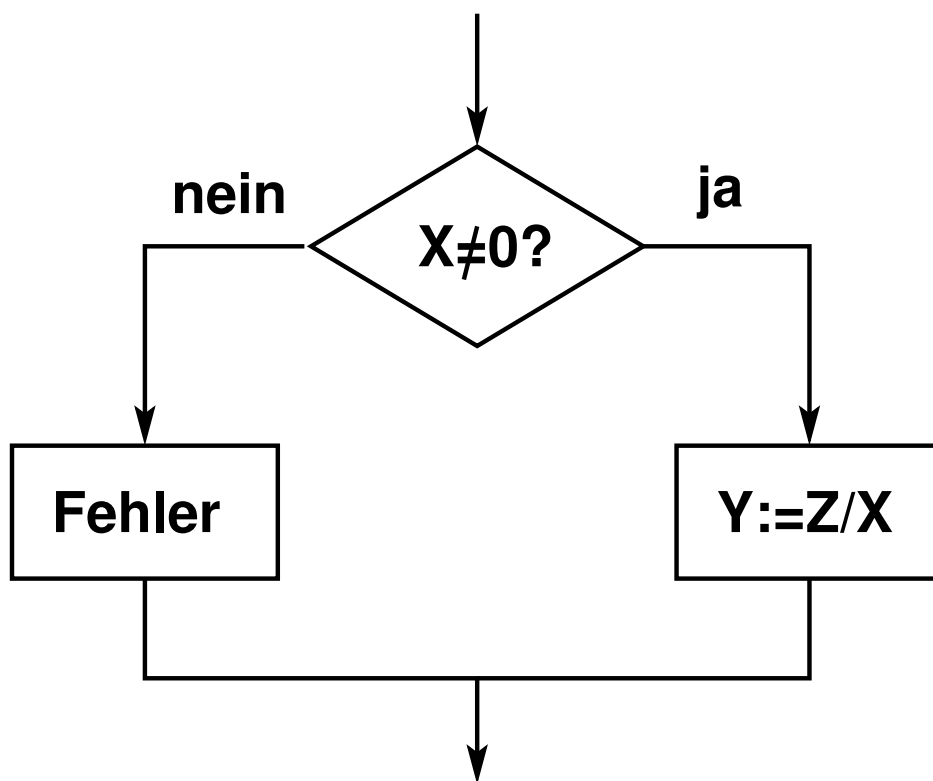
```
#!/bin/bash
```

```
head -2 $1 ; tail -n +3 $1 | sort -k $2 -n
```

```
$ hsort2.sh planeten2.txt 2
```

# Fallunterscheidungen

## Motivation



# Fallunterscheidungen

## Bedingte Ausführung

Bedingte Ausführung: if ... then ... else  
oder auf Deutsch: wenn ... dann ... sonst

Wenn diese **Bedingung** erfüllt ist...

```
if test $1 = "eins"
then
    echo "$1 ist gleich eins"
else
    echo "$1 ist ungleich eins"
fi
```

dann mache dies  
(Bedingung **erfüllt** )

sonst ( **nicht erfüllt** ) mache das

Bei = auf Leerzeichen achten: `=`



# Fallunterscheidungen

Beispiel: Funktionalität von hsort und hsort2 zusammenfassen

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

```
#!/bin/bash
```

```
if test $# = 1 # 1 Argument?
```

```
then # ja, Aufruf wie hsort
```

```
head -2 $1 ; tail -n +3 $1 | sort
```

```
else # nein, Aufruf wie hsort2
```

```
head -2 $1 ; tail -n +3 $1 | sort -k $2 -n
```

```
fi
```

# Fallunterscheidungen

## Vorhandensein einer Datei als Bedingung

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

```
if test -r $1
then
    echo "Die Datei $1 ist vorhanden und lesbar!"
else
    echo "Schade, $1 kann nicht geöffnet werden!"
fi
```

# Fallunterscheidungen

## Fehlertolerante Skripte

```
if test ! -r $1      Negiert die Bedingung ("wenn nicht ...")
then
    echo "Die Datei $1 ist nicht vorhanden!"
    exit 1
fi
```

← Bricht das Skript an dieser Stelle ab

```
if test $# = 1
then
    head -2 $1; tail +3 $1 | sort
else
    head -2 $1; tail +3 $1 | sort -k $2 -n
fi
```

# Fallunterscheidungen

Schreibweisen; test und []

## Gesamter if-Ausdruck in einer Zeile:

```
if test -r datei.txt; then echo da; else echo fehlt; fi
```

- ▶ Ausdruck durch Semikolon trennen!

## test hat zwei äquivalente Schreibweisen:

```
if test -r datei.txt; then ...
```

```
if [-r datei.txt]; then ...
```

- ▶ bei der [...] -Variante auf die Leerzeichen ␣ achten!

Wegen der besseren Lesbarkeit zeigen wir hier die erste Schreibweise.

(Real sieht man oft die zweite, da kürzer)

# Fallunterscheidungen

(wichtigste) Testmöglichkeiten mit Zeichenketten

`test "$1" = "hallo"`    `#` Zeichenkette gleich "hallo"?

`test "$1" != "hallo"`    `#` Zeichenkette nicht "hallo"?

`test -z "$1"`    `#` \$1 ist die leere Zeichenkette

`test -n "$1"`    `#` \$1 ist nicht die leere Zeichenkette

(Nur) bei Zeichenketten: Variablen in "" setzen ("`$1`")

▶ sonst Syntaxfehler wenn Variable = leere Zeichenkette

# Fallunterscheidungen

## Testmöglichkeiten mit Ganzzahlen

<code>test \$1 -eq 42</code>	<code># Zahl = 42?</code>	<i>equal</i>
<code>test \$1 -gt 42</code>	<code># Zahl &gt; 42?</code>	<i>greater than</i>
<code>test \$1 -ge 42</code>	<code># Zahl <math>\geq</math> 42?</code>	<i>greater or equal</i>
<code>test \$1 -lt 42</code>	<code># Zahl &lt; 42?</code>	<i>less than</i>
<code>test \$1 -le 42</code>	<code># Zahl <math>\leq</math> 42?</code>	<i>less or equal</i>
<code>test \$1 -ne 42</code>	<code># Zahl <math>\neq</math> 42?</code>	<i>not equal</i>

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

# Fallunterscheidungen

(wichtigste) Testmöglichkeiten bzgl. Dateien

```
test -f $1  # Datei $1 existiert und ist reguläre Datei
```

```
test -r $1  # Datei $1 existiert und ist lesbar
```

```
test -w $1  # Datei $1 existiert und ist schreibbar
```

```
test -x $1  # Datei $1 existiert und ist ausführbar
```

```
test -d $1  # $1 existiert und ist ein Verzeichnis
```

- ▶ es gibt noch mehr tests
- ▶ siehe man test

# Fallunterscheidungen

logisches UND zwischen zwei Vergleichen

```
if test "$1" = "rot" && test "$2" = "blau" ; then
    echo wahr
else
    echo falsch
fi
```

```
$ ./skript.sh rot blau
wahr
$ ./skript.sh gruen blau
falsch
$ ./skript.sh rot gruen
falsch
$ ./skript.sh blau rot
falsch
```

Wahrheitstabelle log. UND:

<u>wahr</u> $\wedge$ <u>wahr</u>	=	<u>wahr</u>
<u>wahr</u> $\wedge$ falsch	=	falsch
falsch $\wedge$ <u>wahr</u>	=	falsch
falsch $\wedge$ falsch	=	falsch



# Fallunterscheidungen

logisches ODER zwischen zwei Vergleichen

```
if test "$1" = "rot" || test "$2" = "blau" ; then
    echo wahr
else
    echo falsch
fi
```

```
$ ./skript.sh rot blau
wahr
$ ./skript.sh rot xxx
wahr
$ ./skript.sh xxx blau
wahr
$ ./skript.sh xxx xxx
falsch
```

Wahrheitstabelle log. ODER:

<u>wahr</u> ∨ <u>wahr</u>	=	<u>wahr</u>
<u>wahr</u> ∨ falsch	=	<u>wahr</u>
falsch ∨ <u>wahr</u>	=	<u>wahr</u>
falsch ∨ falsch	=	falsch

# Fallunterscheidungen

Komplexe Ausdrücke über UND und ODER

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

```
if test "$1" = "rot" &&  
    (test "$2" = "apfel" || test "$2" = "kirsche") ; then  
    echo wahr  
else  
    echo falsch  
fi
```

```
$ ./skript.sh rot apfel  
wahr  
$ ./skript.sh rot kirsche  
wahr  
$ ./skript.sh rot banane  
falsch  
$ ./skript.sh gruen apfel  
falsch
```

$\underline{W} \wedge (\underline{W} \vee \underline{W})$	=	$\underline{W}$
$\underline{W} \wedge (\underline{W} \vee \underline{F})$	=	$\underline{W}$
$\underline{W} \wedge (\underline{F} \vee \underline{W})$	=	$\underline{W}$
$\underline{W} \wedge (\underline{F} \vee \underline{F})$	=	$\underline{F}$
$\underline{F} \wedge (\underline{W} \vee \underline{W})$	=	$\underline{F}$
$\underline{F} \wedge (\underline{W} \vee \underline{F})$	=	$\underline{F}$
$\underline{F} \wedge (\underline{F} \vee \underline{W})$	=	$\underline{F}$
$\underline{F} \wedge (\underline{F} \vee \underline{F})$	=	$\underline{F}$

# Fallunterscheidungen

## Wahrheitswerte von Kommandos

Kommandos wie `grep` und `diff` haben Wahrheitswerte:

`grep wort datei`     `true`  $\Leftrightarrow$  `datei` enthält `wort`

`diff datei1 datei2`     `true`  $\Leftrightarrow$  `datei1` und `datei2` sind gleich

Beispiele:

```
if grep hallo datei.txt; then ...
```

```
if echo $1 | grep hallo; then ...
```

u.s.w.

- ▶ bei *Pipes* gilt der Wert des letzten Befehls

# Fallunterscheidungen

## Wahrheitswerte von Kommandos

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte  
Wiederholung

Verzweigungen

if  
test  
grep

Variablen

case

```
#!/bin/bash
```

```
if grep -q -i $1 planeten.txt; then  
    echo $1 ist ein Planet  
else  
    echo $1 ist kein Planet  
fi
```

---

```
$ ./skript.sh erde  
erde ist ein Planet  
$ ./skript.sh pluto  
pluto ist kein Planet
```

grep -q: quiet; unterdrückt Ausgabe von grep

# Variablen

## Variablenzuweisungen

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test


grep

Variablen

case

Wert an Variablen zuweisen:

keine Leerzeichen!

  
`$ wort=eins`

Variablenwert benutzen / ausgeben:

```
$ echo $wort
```

```
eins
```

# Variablen

## Datentypen

Variablen sind “schwach getypt”

- ▶ werden automatisch als Zeichenkette oder Zahl benutzt

Beispiel:

```
$ a=1
```

```
$ b="2"
```

```
$ if test $a -lt $b; then echo wahr; fi  
wahr
```

```
$ if test $a = "1"; then echo wahr; fi  
wahr
```

```
$ if test $a -lt "zwei"; then echo wahr; fi  
bash: test: zwei: Ganzzahliger Ausdruck erwartet.
```

# Variablen

## Variablenzuweisungen aus Shell-Aufrufen

Zwischenspeichern von Programmausgaben:

```
$ a=$(echo -n Linux | wc -m)
$ echo $a
5
```

Auch eine komplette Zeile kann man sinnvoll speichern:

```
$ a=$(ls -l eins.txt)
$ echo $a
-rw-r--r-- 1 cg stud 4502 17. Nov 16:38 eins.txt
```

► Mehrzeilige Ausgaben besser nicht in Variablen packen!

# Variablen

Variablen als Zeichenketten verarbeiten

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

```
$ name=datei
$ verz=/home/juser
$ pfad=$verz/$name.jpg
$ echo $pfad
/home/juser/datei.jpg
```



# Variablen

## Variablen als Zeichenketten: Sonderfälle

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

Variablennamen durch Klammern vom Text abtrennen:

```
$ name=zeichen  
$ echo ${name}kette  
zeichenkette
```

Leerzeichen durch Anführungszeichen ("...") erhalten:

```
$ a=eins  
$ b=zwei  
$ c="$a $b"  
$ echo $c  
eins zwei
```

# Variablen

## Variablen und Arithmetik

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

### `$((...))`: Arithmetischen Ausdruck auswerten

```
$ echo $((3+5))  
8
```

### Mit Variablen:

```
$ a=9  
$ b=3  
$ echo $((a*b))  
27
```

- ▶ Im arithm. Ausdruck darf man `a` statt `$a` schreiben

# Variablen

## Arithmetische Operatoren

---

+	Addition
-	Subtraktion
*	Multiplikation
/	ganzzahlige Division
%	Modulo (Rest der Division)
**	Potenz

---

Beispiel:

```
$ echo $((23 / 5))  
4  
$ echo $((23 % 5))  
3
```

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

# Variablen

## Zufallszahlen und Modulo

- ▶ Shell kann nur Ganzzahlen verarbeiten
- ▶ Fließkommazahlen: z.B. mit `$(bc -l)`

`$RANDOM`: liefert Zufallszahl zwischen 0...32767

Würfel mit 6 Seiten simulieren:

```
$ echo $((1+RANDOM%6))
```

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

# Mehrfache Fallunterscheidungen

case

## case: Mehrfache Fallunterscheidung

```
case $1 in
    null) echo 0
    ;;
    eins) echo 1 ← Befehl(e) die bei $1=eins
                  ausgeführt werden sollen
    ;;
    ...
    *) echo "$1 ist keine Ziffer"
    ;;
esac
```

Entscheidungsvariable

potentieller Wert von \$1

Ende dieses Falls

Auffangebene:  
alle vorherigen Fälle unzutreffend

# Mehrfache Fallunterscheidungen

Lösung mit if

Würfel mit Kommentar:

```
#!/bin/bash

wurf=$((1+RANDOM%6))

if test $wurf = 1; then
    echo "Oh je, eine 1"
else
    if test $wurf = 6; then
        echo "Juhu, eine 6"
    else
        if test $wurf = 5; then
            echo "Gut, eine 5"
        else
            echo "Eine $wurf"
        fi
    fi
fi
```

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

# Mehr Elemente von Shellskripten

Lösung mit case

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

```
#!/bin/bash
```

```
wurf=$((1+RANDOM%6))
```

```
case $wurf in
    1)    echo "Oh je, eine 1"
        ;;
    6)    echo "Juhu, eine 6"
        ;;
    5)    echo "Gut, eine 5"
        ;;
    *)    echo "Eine $wurf"
        ;;
esac
```

# Überblick

- ▶ `if...then...else`
- ▶ `test` (auf Gleichheit, Vorhandensein einer Datei...)
- ▶ `grep -q` (für `test`, “quiet”)
- ▶ `$((...))` Arithmetische Ausdrücke auswerten
- ▶ `case` (wie `if`, mehr Fälle)
- ▶ Variablen: z.B. `wurf=...`, `$wurf`
- ▶ Umgebungsvariable `RANDOM`



# Nächstes Mal

- ▶ `for`-Schleifen
- ▶ `seq`
- ▶ `csv`-Tabellen
- ▶ `cut` and `tr` (`trim`)

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

`if`

`test`

`grep`

Variablen

`case`

# Ende der heutigen Vorlesung

Linux-  
Praktikum

Dirk Frettlöh

Shellskripte

Wiederholung

Verzweigungen

if

test

grep

Variablen

case

**Vielen Dank fürs Zusehen!**

**Bis nächste Woche!**