

# **Unix – ganz schmerzlos...**

## Teil 8 - Shell-Skripte III

Jörn Clausen

jc@Genetik.Uni-Bielefeld.DE

# weitere Kontrollstrukturen

- Mehrfachverzweigung:

```
case $1 in
  1)
    echo I ;;
  2)
    echo II ;;
  3)
    echo III ;;
  4)
    echo IV ;;
  5)
    echo V ;;
  *)
    echo Zahl zu gross
esac
```

- pattern matching:

```
case $monat in
  *r*)
    echo Schneegefahr ;;
  *)
    echo Schnee sehr unwahrscheinlich
esac
```

# erweiterte `if`-Verzweigung

- mehrere Abfragen:

```
if [ "$1" = "hallo" ]; then
    echo auch hallo
elif [ "$1" = "tschuess" ]; then
    echo auf wiedersehen
elif [ -z "$1" ]; then
    echo wie bitte? sprich etwas lauter
else
    echo $1? das verstehe ich nicht
fi
```

- beliebige Tests
- aber: sinnvolle Zusammenfassungen

# Rechnen

- Shell hat keine arithmetischen Funktionen

- externes Programm: `expr`

```
juser@hobel> expr 5 + 7 \* 2  
19
```

- Verwendung in Skripten: backtick evaluation

```
num=1  
while [ $num -le 10 ]; do  
    sq=`expr $num \* $num`  
    echo $num zum Quadrat ist $sq  
    num=`expr $num + 1`  
done
```

# Analyse von Dateinamen

- Verzeichnisse abschneiden:

```
juser@hobel> basename /homes/juser/foo.jpg  
foo.jpg
```

- Suffix abschneiden:

```
juser@hobel> basename /homes/juser/foo.jpg .jpg  
foo
```

- Dateiname abschneiden:

```
juser@hobel> dirname /homes/juser/foo.jpg  
/homes/juser
```

# Beispiel: Bilder konvertieren

- unterschiedliche Dateiformate lesen:

```
case $file in
  *.gif)
    name=`basename $file .gif`
    prog="giftopnm"
    ;;
  *.jpg)
    name=`basename $file .jpg`
    prog="djpeg"
    ;;
  *)
    echo unknown format
esac
```

- Fortsetzung in den Übungsaufgaben

# Ausblick

- Shell-Skripte geeignet für
  - kleine Probleme
  - ad-hoc-Lösungen
  - wenn für Teilprobleme Tools existieren
- weniger geeignet für
  - komplexe Probleme
  - neue Funktionalität
- teuer: externe Programme starten
- bei größeren Aufgaben: Perl, C, Java, ...