Computer & GNU/Linux Einführung
Teil 4

Simon Haller, Sebastian Stabinger, Benjamin Winder

Inst. für Informatik
[IFI]

September 26, 2013

Shell–Skripte

Struktur eines einfachen Shell–Skriptes

```
#!/bin/sh      # legt das Programm mit dem das Skript
               # ausgeführt wird fest.
# Kommentar (z.B. Autor und Erklärung zum Programm)
Befehl 1
Befehl 2
```

Shell–Skripte

Bedingte Abfrage

```
#!/bin/sh
# wenn dann Konstrukt
 if [BEDINGUNG]; then
    # DO SOMETHING
   elif [ANDERE-BEDINGUNG]; then
    # DO SOMETHING
    else
    # DO SOMETHING
 fi
```

Vergleichsoperationen

Dateien

```
  -f                    reguläre Datei exisitert
  -d                    Verzeichnis exisitert
  -r                    Datei exisitert und ist lesbar
  Datei1 -nt Datei2     wahr, wenn Datei1 neuer als die
Datei2
```

Vergleichsoperationen

Zeichenketten

```
"STR1" = "STR2"    wahr, wenn die Zeichenketten gleich sind
"STR1" != "STR2"   wahr, wenn die Zeichenketten ungleich
sind
-z "STR1"          wahr, wenn die Zeichenkette leer ist
-n "STR1"          wahr, wenn die Zeichenkette nicht leer ist
```

Vergleichsoperationen

Zahlen

```
n1 -eq n2  wahr, wenn die Zahlen gleich sind
n1 -ne n2  wahr, wenn die Zahlen ungleich sind
n1 -gt n2  wahr, wenn die Zahl n1 grer ist als n2
n1 -ge n2  wahr, wenn die Zahl n1 grer oder gleich n2 ist
```

Sonstiges

```
!    Negation
```

Shell–Skripte

Beispiel

```
#!/bin/sh
VAR1="foo"
VAR2="bar"
if [ "$VAR1" = "$VAR2" ]; then
  echo die Auswertung des Ausdruck ist Wahr
else
  echo die Auswertung des Ausdruck ist Falsch
fi
```

Aufgaben

Aufgabe 1

- Erstelle eine Datei die nur die Namen der Benutzer die ein 'share'-Verzeichnis besitzen beinhaltet.

Aufgabe 2

- Erstelle ein Skript, das dir Sonntags sagt 'du sollst nicht arbeiten'.

Mögliche Lösungen

Aufgabe 1: `find`

```
find /afs/zid1.uibk.ac.at/shares/*/* -maxdepth 0 -exec basename {} \; >
/tmp/mfile
```

Aufgabe 2: Shell–Skript

```
#!/bin/sh
# errechnen des wochentags (sonntags ... 0)
WEEKDAY="$(date +%w)"
# wenn Sonntag dann echo.
if [ $WEEKDAY -eq "0" ]; then
  echo "Du sollst nicht arbeiten."
fi
```

Kommandos

seq [START] [ENDE] *(sequence)*

gibt die Zahlen von [start] bis [ende] aus.
Eine Schrittweite kann:

```
$ seq [start] [schrittweite] [ende]
```

angegeben werden

Kommandos

seq [START] [ENDE] *(sequence)*

gibt die Zahlen von [start] bis [ende] aus.
Eine Schrittweite kann:

```
$ seq [start] [schrittweite] [ende]
```

angegeben werden

Beispiel

```
seq 1 5
```

liefert:

```
1
2
3
4
5
```

Einfache Schleifen

for Schleife

```
for Variable in [Liste] ; do
  etwas mit Variable
done
```

Einfache Schleifen

for Schleife

```
for Variable in [Liste] ; do
  etwas mit Variable
done
```

Beispiel

```
for i in $(seq 1 10)
  do echo $i
done
```

Einfache Schleifen

Beispiel in Dos

```
REN *.FOR *.F
```

Einfache Schleifen

Beispiel in Dos

```
REN *.FOR *.F
```

Aufgabe

löse dieses Problem in einer Shell
Tipp: funktioniert nur mit einer for-Schleife

Einfache Schleifen

Beispiel in Dos

```
REN *.FOR *.F
```

Aufgabe

löse dieses Problem in einer Shell
Tipp: funktioniert nur mit einer for-Schleife

mögliche Lösung in einer Shell

```
for i in *.for; do
  mv $i $(basename $i .for).f
done
```

Fink (2006)

Einfache Schleifen

while Schleife

```
while [ Bedingung ] ; do
  kommandos
done
```

Einfache Schleifen

while Schleife

```
while [ Bedingung ] ; do
  kommandos
done
```

until Schleife

```
until [ Bedingung ] ; do
  kommandos
done
```

Einfache Schleifen

while Schleife

Beispiel:

```
while true ; do
  echo Hallo
done
```

Shellvariablen

in Skripten

- "$0" Name des Skriptes
- "$1" -"$9" Übergabeparameter an Stelle n
- "$@" equivalent zu "$1" "$2"...
- "$*" ähnlich "$@" mit der IFS Variable als Trennzeichen
- "$#" Zahl der positionellen Parameter

Beispiel

Aufgabe

- schreibe ein Skript, das zuerst den Skriptnamen und dann zeilenweise die Übergabeparameter ausgibt.

Beispiel

Aufgabe

- schreibe ein Skript, das zuerst den Skriptnamen und dann zeilenweise die Übergabeparameter ausgibt.

mögliche Lösung

```
#!/bin/sh
echo "$0"
for i in "$@" ; do
   echo "$i"
done
```

Webseiten mit cat lesen

```
#/bin/bash
   # Filedescriptor auf tcp/WEBSEITE
   exec 5<>/dev/tcp/$1/80
   # sende mit echo einen HTTP GET request
   # auf den vorbereiteten Descriptor
   echo -e "GET / HTTP/1.0\nHost:  $1 \n" >&5
   # Lese die Antwort des Servers
   cat <&5
   # Liste der Filedescriptoren
   # ls -l /proc/self/fd
   # Schliessen des Filedescriptors
   exec 5<&-
```

Überblick Teil 1

Kommando	**Bedeutung**
alias	Kommandosequenz zusammenfassen
cat	Dateien ausgeben/ aneinanderhängen
cd	Verzeichnis wechslen
chmod	Dateizugriffsrechte ändern
chown	Owner ändern
cp	Dateien kopieren
echo	anzeigen von einer Textzeile
head	Kopf einer Datei ausgeben
ln	Dateien verlinken
ls	Verzeichnis auflisten
mkdir	Verzeichnis erstellen
mv	Dateien verschieben / umbenennen
rm	Dateien löschen
set, unset	setzen und löschen von Variablen
tail	Ende einer Dateien ausgeben
type	abfragen welches Kommando ausgefürt wird

Überblick Teil 2

Kommando	**Bedeutung**
basename	Name des Skriptes
bc	Rechner
dirname	Name des Verzeichnisses
for Var in List; do etwas; done	Schleife
grep	durchsuche Dateien nach Ausdrücken
if X; then Y; [else] Z; fi	Bedingte Anweisung
more, less	zeigt Dateien seitenweise
sed	Streameditor
sort	Zeilen von Textdateien sortieren
tar	verwaltet Dateiarchive
tee	Splitten von STDOUT
tr	Zeichen übersetzen (translate)
wc	Wörter zählen
while X ; do Y ; done	Schleife

References I

Fink, M. (2006). Unix kurs.