Betriebssysteme

Übungen 03

Prof. Dr. Rainer Werthebach

Studiengang Informatik

Hochschule Aalen - Technik und Wirtschaft

Skripts

- Was ist ein Skript?
- Beispiel: laenge1.sh

```
#!/bin/bash
echo "Die Anzahl Woerter in myfile ist"
cat myfile | wc -w
```

```
Hinweis: Das RRZN-Skript „Unix – eine Einführung“
         ist sehr hilfreich für die Shellskript
         Übung. Erhältlich in der Bibliothek.
```

„Shebang“ (Magic Line, Hash-Bang)

- Erste Zeile eines Skriptes
- In ihr wird der Pfad zum Interpreter festgelegt
- Abstraktes Beispiel:
  - `#!/pfad/zum/interpreter`
- Beispiele:
  - `#!/bin/sh` (Bourne-Shell, oder heute eher Link auf Standard-Shell)
  - `#!/bin/bash` (Bourne Again-Shell)
  - `#!/bin/csh` (C-Shell)
  - `#!/bin/ksh` (Korn-Shell)

Skript aufrufen

- Möglichkeit 1: `sh` *`Skript`* `[Argumente / Parameter]`

- Möglichkeit 2: `./Skript [Argumente / Parameter]`
  - zu beachten: Shellskript muss ausführbar sein
  - d.h. `chmod u+x` *`Skript`* nicht vergessen

- Möglichkeit 3: `.` *`Skript`* `[Argumente / Parameter]`
  - alternativ: `source` *`Skript`* `[Argumente / Parameter]`
  - das Skript wird in der aktuellen Shell und nicht wie oben in einer Subshell (als Kindprozess) ausgeführt

Standard Variablen, positionale Parameter

- Allgemein:
  - `$?` → Rückgabewert des letzten Kommandos, vgl. auch echo $?
  - `$$` → Prozessnummer (PID) der aktiven Shell
  - `$0` → Name des Programms
- Positionale Parameter:
  - `$#` → Anzahl der Aufrufparameter
  - `$n` → Wert des Aufrufparameter n (1 <= n <= 9)
  - `shift 5` → alle Aufrufparameter um 5 verschieben (`$6` jetzt `$1` usw.)
  - `set Wert1 Wert2 Wert3 …` → Werte der Aufrufparameter setzen
  - `$*` → alle Aufrufparameter (als ein String)
  - `$@` → alle Aufrufparameter (als einzelne Strings)

Variablen

- `Variable=Wert`
  - Erzeugen und Setzen einer Variable
- `${Variable}`
  - Inhalt der Variable
  - `{ }` müssen nicht angegeben werden, wenn die Variable von Trennzeichen umgeben ist
- `${Variable:-Wert}`
  - Inhalt der Variable
  - Falls Variable nicht gesetzt ist, wird Wert verwendet
- `${Variable:=Wert}`
  - Inhalt der Variable
  - Falls Variable nicht gesetzt ist, wird Wert verwendet und Variable wird Wert zugewiesen

Variablen (Fortsetzung)

- `${Variable:+Wert}`
  - Verwendet Wert, falls die Variable gesetzt ist, andernfalls nichts
- `${Variable:?Wert}`
  - Wert der Variable
  - Falls die Variable nicht gesetzt ist, wird Wert ausgegeben und die Shell wird beendet.
  - Wenn kein Wert angegeben wurde, wird der Text „parameter null or not set“ ausgegeben

Subshell

- Aufruf:
  - `$(Befehl / Skript)`
  - Oder (eher veraltet mit back ticks): `` `Befehl / Skript` ``
- übergeben werden:
  - alle definierten Variablen
  - alle positionalen Parameter
  - die umask
  - Werte der Sondervariablen (PATH, IFS: „<space><tab><newline>“, PS1, PS2)
- Rückgabe:
  - `anzWoerter=$(wc -w testdatei.txt)`

Einlesen – Dateien, Benutzereingaben

- Kommando: `read`
- Beispiele:
  - `read a` → Lese Benutzereingabe in Variable a ein
  - `read a b c` → Lese Benutzereingabe (getrennt durch Leerzeichen) in Variablen ein
  - `read a < datei.txt` → Lese erste Zeile der Datei datei.txt ein
  - `read a b c < datei.txt` → Lese von erster Zeile erstes Wort in a, zweites Wort in b, den Rest der Zeile in c
  - `{ read a; read b; read c; } < datei.txt` → Lese Datei Zeilenweise ein (ersten 3 Zeilen, Rest wird ignoriert)

Here Document

- Für mehrzeilige Eingaben für Kommandos
- Syntax:
  - ```
    Kommando << Trennstring
    Eingabe
    ...
    letzte Eingabe
    Trennstring
    ```
    ```

Here Document - FTP Beispiel

```
#!/bin/bash
Server="ftp.infotronik.htw-aalen.de"
Benutzer="user"
Password="12345"
Directory="Oeffentlicher-Ordner"
ftp -n $Server << FTP_EINGABE
user $Benutzer "$Password"
binary
cd $Directory
put "testdatei"
bye
FTP_EINGABE
exit 0
```

for - Anweisung

- Syntax:
  - ```
    for Variable in Wörterliste
    do
      Kommandoliste
      ...
    done
    ```
- Beispiel:
  - ```
    for Eintrag in $(ls -1 ~)
    do
      echo "--> $Eintrag"
    done
    ```
- Falls „`in Wörterliste`“ nicht angegeben ist, wird die Parameterliste (`$@`) verwendet