# Betriebssysteme 2 BS2-P

**Foliensatz P**

**Hinweise zum Praktikum**

---

Prof. Dr.-Ing. Hans-Georg Eßer

esser.hans-georg@fh-swf.de
http://swf.hgesser.de/

27. Oktober 2016

FH Südwestfalen, Informatik, WS 2016/17 v1.0, 2016/10/27

## Bearbeitung der Skript-Aufgaben (1)

Bitte die Hinweise im Dokument „Bash Style Guide + Kodierungsrichtlinie“ [Meh14] beachten → `https://lug.fh-swf.de/vim/vim-bash/StyleGuideShell.de.pdf`

- Exit-Codes externer Programme prüfen

```
function fatal() {
  echo -e "$0: fatal error: $1\nExiting"
  exit 1
}

dir=./tmpdir
mkdir ${dir} || fatal \
  "Kann Verzeichnis ${dir} nicht erzeugen"
```

- Prüfung bei jeder Aktion, die schief gehen kann, nötig



## Bearbeitung der Skript-Aufgaben (2)

- Parameter auf Anzahl **und** Validität prüfen, z. B.
  - Falsche Parameterzahl?

```
[ $# -ne 3 ] && fatal "brauche drei Argumente"
```

  - Parameter ist Zahl?

```
numbers='^[0-9]+$'
[[ $var =~ $numbers ]] || fatal "keine Zahl"
```

  - Parameter ist zu lang?

```
[ ${#var} -gt 10 ] && fatal "zu lang"
```

  - (… oder mit **if** … **then**-Syntax, wenn Sie die &&- und ||-Varianten nicht mögen)



## Bearbeitung der Skript-Aufgaben (3)

- Obige Beispiele sind genau das: *Beispiele* – also denkbare Fehler mit möglichen Reaktionen (dort immer: Abbruch)
- Allgemein: Fehler entdecken ist nur der erste Schritt
  1. Welche Ursachen kann der Fehler haben? Alle potenziellen Fehlerursachen auflisten und angemessen behandeln
  2. Wenn kein Skript-Abbruch erfolgt, was dann?
- zum obigen `mkdir`-Beispiel:
  - Mögliche Ursache: Verzeichnis schon vorhanden. Ist das Scheitern von `mkdir` dann ein Fehler? Wenn nicht, ggf. auf `mkdir -p` umstellen
  - Mögliche Ursache: fehlende Schreibrechte im Zielordner → wird Verzeichnis im Folgenden dringend gebraucht?
  - Was tun, wenn eines von mehreren `mkdir`-Kommandos (Schleife) fehlschlägt?

## Bearbeitung der Skript-Aufgaben (4)

- Code kommentieren; vgl. [Meh14, Abschnitt 3]
  - auch: Funktions- und Abschnittskommentare
- Zum Kommentieren gehört auch:
  - sinnvolle („sprechende“) Variablen- und Funktionsnamen
  - Langoptionen (`--verbose` statt `-v`)



## Bearbeitung der Skript-Aufgaben (5)

Vorsicht bei Befehlen, die vom erfolgreichen Ausführen eines vorhergehenden Befehls abhängen!

- Beispiel (gestern im Praktikum entdeckt): Verzeichniswechsel

```
dirlist=$( find . -type d | grep -v "^\.$")
for dir in $dirlist; do
  cd $dir           # in Unterordner wechseln
  ; irgendwas in $dir machen
  cd ..             # zurück nach oben
done
```

Sehen Sie hier ein Problem?



## Bearbeitung der Skript-Aufgaben (5)

Vorsicht bei Befehlen, die vom erfolgreichen Ausführen eines vorhergehenden Befehls abhängen!

- Beispiel (gestern im Praktikum entdeckt): Verzeichniswechsel

```
dirlist=$( find . -type d | grep -v "^\.$")
for dir in $dirlist; do
  cd $dir           # in Unterordner wechseln
  ; irgendwas in $dir machen
  cd ..             # zurück nach oben
done
```

Sehen Sie hier ein Problem?

- Ordner: `"abc"`, `"def"`, `"gh␣ij"` (mit Leerzeichen)
- `find` sucht rekursiv – nach Wechsel in `abc/def` ist `cd ..` falsch! → Lösung: `pushd` und `popd`



## Bearbeitung der Skript-Aufgaben (6)

- **Ab Übungsblatt 4 zudem nötig: Testprotokoll**
  - Test-Cases (Standard- und Fehlersituationen)
  - erwartetes Verhalten, beobachtetes Verhalten
  - Protokoll kann in einfache Textdatei geschrieben werden, keine *fancy*-Formatierung o. ä. nötig
  - wo möglich: direkt Protokoll-Feature in Skript einbauen, z. B. Option `-v`, die ausführliche Info nach `programm.log` schreibt
- Allgemein: Für Shell-Programmierung gelten gleiche Anforderungen wie für reguläre Software-Entwicklung
- Orientieren Sie sich **nicht** an den Scripts aus der Vorlesung:
  - Rapid Prototyping
  - darum unkommentiert, meist ohne Fehlerfall-Prüfung
  - bei kleineren Projekten ein guter Anfang, danach sauber kommentieren und Fehler abfangen

## Erinnerung zu Testaten

- Abgabe ist **zu Beginn** der Prakt.-Stunde
- Lösungen müssen dann vollständig sein