

Shell (bash) Einführung

Shell (bash) Introduction

Mag. Thomas Griesmayer

Zweck der Shell

- Sie ist eine Schnittstelle zwischen Benutzer und Unix.
- Bei den meisten Unix-Implementierungen gibt es eine Vielzahl an Shells.
- Interaktive Arbeit (im Dialog) - Das System erwartet die Eingabe eines Befehls.
- Variablen - Eine Unix-Shell definiert Variablen zur Steuerung des Systemverhaltens.
- Shell Scripts - Eine Reihe spezieller Befehle dient zur Erstellung von Programmen.

Shell Arten

- Die Bourne- oder Standardshell ist die kompakteste und einfachste Form.
- Die Korn-Shell, eine Weiterentwicklung der Bourne-Shell, erlaubt das Editieren in der Befehlszeile.
- C-Shell, die mit der Syntax von C arbeitet, bietet einige Annehmlichkeiten.
- Die bash (Bourne Against Shell) ist jene Shell auf die wir näher eingehen.
- ...

Variablen anzeigen

- Eine Liste der Variablen und deren Werte erhält man durch Aufruf des Befehls „set“.
- Eine Variable kann man mit dem \$ Zeichen ansprechen.
- Sollten sich im Variablennamen Sonderzeichen befinden, kann man diese über \${...} ansprechen.

```
$ set
HUSHLOGIN=FALSE
SHELL=/bin/bash
$ echo SHELL
SHELL
$ echo $SHELL
/bin/bash
$ echo ${SHELL}
/bin/bash
```

Arbeitsweise der bash (1)

- Das erste Wort der Eingabe ist der Befehl.
- Befehle, Optionen und Argumente werden durch Trennzeichen (Tabulator oder Leerzeichen) voneinander getrennt.
- Die bash sucht nach Zeichen, die eine besondere Bedeutung haben, wie beispielsweise die Zeichen „*“ oder „?“ . Diese Zeichen werden von der bash interpretiert.

```
$ cal date
cal: illegal year value: use 1-9999
$ ls datei?.txt b*
berechnung.txt  datei1.txt      datei2.txt
```

Arbeitsweise der bash (3)

- Bereitstellen des Prompts und warten auf Eingabe (Befehlszeile).
- Durchsuchen der Befehlszeile nach Sonderzeichen und Auswertung dieser Sonderzeichen („*“, „?“, „>“, „\$“).
- Feststellen des ersten Wortes der Eingabe.
- Ausführen des Befehls (=erstes Wort der Eingabe).

```
$ ls da* >inhalt_da.txt
$ ls datei1.txt datei2.txt >inhalt_da.txt
$ cat inhalt_da.txt
datei1.txt
datei2.txt
```

Argumente (1)

Typ	Beschreibung
arg	Das Argument wird als Text betrachtet.
\char backslash	Das Fluchtsymbol entwertet das unmittelbar folgende Sonderzeichen. <code>\$ echo *</code>
"arg" doublequote	Alle von doppelten Hochkomma eingeschlossenen Zeichen verlieren ihre Sonderfunktion. Ausnahme: <code>\$'\`</code> <code>\$ echo "\$SHELL"</code>
'arg' quote	Alle von einfachen Hochkomma eingeschlossenen Zeichen verlieren ihre Sonderfunktion. <code>\$ echo '\$SHELL'</code>
`cmd` backquote	Das Argument ist das Ergebnis (Ausgabe) der Ausführung des Kommandos. <code>\$ echo `logname`</code>
\$(arg)	Das Argument ist das Ergebnis (Ausgabe) der Ausführung des Kommandos. <code>\$ echo working directory \$(pwd)</code>
\${}	Das Argument ist der Ausdruck, welcher berechnet (Arithmetic Expression) wird. <code>\$ echo \${2+3}</code>
\$(())	Das Argument ist der Ausdruck, welcher berechnet (Arithmetic Expression) wird. <code>\$ echo \$((2+3))</code>