BASH - Tipps und Tricks

Brace Expansion / Klammern Expansion

Klammern Expansion kann einem das Leben leichter machen.

Beispiele:

- Anlegen mehrerer “gleicher” Verzeichniss-Bäume

```
mkdir -p {ebene1dir1,ebene1dir2}/{sub1,sub2,sub3}
```

Legt folgende Verzeichniss-Struktur an:

```
ebene1dir1:
total 20
drwxr-xr-x  5 root root 4096 2009-06-03 15:48 .
drwxrwxrwt 14 root root 4096 2009-06-03 15:48 ..
drwxr-xr-x  2 root root 4096 2009-06-03 15:48 sub1
drwxr-xr-x  2 root root 4096 2009-06-03 15:48 sub2
drwxr-xr-x  2 root root 4096 2009-06-03 15:48 sub3

ebene1dir2:
total 20
drwxr-xr-x  5 root root 4096 2009-06-03 15:48 .
drwxrwxrwt 14 root root 4096 2009-06-03 15:48 ..
drwxr-xr-x  2 root root 4096 2009-06-03 15:48 sub1
drwxr-xr-x  2 root root 4096 2009-06-03 15:48 sub2
drwxr-xr-x  2 root root 4096 2009-06-03 15:48 sub3
```

- Quelle: http://mywiki.wooledge.org/BashGuide/TheBasics/Patterns

Then, there is Brace Expansion. Brace Expansion technically does not fit in the category of Globs, but it is similar. Globs only expand to actual filenames, where brace expansion will expand to any permutation of the pattern. Here's how they work:

```
    $ echo th{e,a}n
    then than
    $ echo {/home/*,/root}/.*profile
    /home/axxo/.bash_profile /home/lhunath/.profile /root/.bash_profile
/root/.profile
    $ echo {1..9}
    1 2 3 4 5 6 7 8 9
    $ echo {0,1}{0..9}
    00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19
```

- Good Practice:
  You should always use globs instead of ls (or similar) to enumerate files. Globs will always

expand safely and minimize the risk for bugs. You can sometimes end up with some very weird filenames. Generally speaking, scripts aren't always tested against all the odd cases that they may end up being used with.

Inhalt einer Datei OHNE KOMMENTARE ausgeben

```
grep -v -e '^[[.:space:]]*#' -e '^$' file.txt
```

$HISTIGNORE - Aufzeichnen einzelner Befehle in der Bash History verhindern

Die BASH besitzt bzw überwacht u.a. die Umgebungsvariable $HISTIGNORE, welche eine durch DOPPELPUNKT getrennte Liste von String-Mustern enhalten kann. Befehle oder Kommandozeilen welche auf eine der Mustern zutreffen werden dann NICHT in die Bash-History aufgenommen.

Durch geschickte Konfiguration der Umgebungsvariablen lässt sich so z.B. einbauen dass Kommandos welche mit einem <SPACE> beginnen grundsätzlich NICHT aufgezeichnet werden sollen. Dies macht es sehr bequem die Aufzeichnung in Einzelfällen auf der Kommandozeile welche z.b. Klartext-Passwörtern enthalten zu unterdrücken.

Beispiel:

Im folgenden Beispiel wurden DREI String-Muster unterdrückt:

```
export HISTIGNORE="[ \t]*:history:history[ \t]"
```

1. unterdrückt alle Zeilen welche mit SPACE oder TAB beginnen
2. unterdrückt nur das alleinstehende “history” Kommando
3. unterdrückt das alleinstehende “history” Kommando mit angehängtem SPACE bzw TAB.

Alle anderen Kommandos, selbst “history 5” usw werden nach wie vor in der Bash-History aufgezeichnet.

In Debian 5 Lenny ist die Umgebungsvariable nicht gesetzt. Die “export” Zeile lässt sich leicht in eine persönliche ~/.bashrc einbauen. z.B.

```
export HISTIGNORE="$HISTIGNORE:[ \t]*:history:history[ \t]"
```

— *Axel Werner 2012-04-05 18:15*

bash, linux, tips, tricks, scripting, cli, shell, history, histignore, ignore, expansion, brace