Tips & Tricks für die Shell

Klaus Knopper <shelltricks@knopper.net>

Version 1.1 (29. August 2000)

Die folgenden Kontrollsequenzen sind Standard auf den meisten Unix-Installationen und können mit dem Kommando `stty` verändert werden.

Beispiele: `stty -a` (zeigt aktuelle Terminal-Einstellungen an)
`stty erase '^?'` (ändert die Zeichen-Löschsequenz auf DEL)

Wichtige Kontrollsequenzen

(^ steht für <Control> bzw. <Strg>)

TAB	Automatische Kommando/Dateinamenergänzung
^C	Abbruch des laufenden Programms (cancel)
^Z	Anhalten des laufenden Programms (suspend)
^U	Löschen der Zeile
^H	Löschen des Zeichens vor dem Cursor (erase)
^D	1. Löschen des Zeichens unter dem Cursor (delete) 2. Beenden der Shell oder der Eingabe (end of file)
^L	Löschen oder Auffrischen des Bildschirms
^S	Anhalten der Ausgabe (scroll lock)
^Q	Wiedereinschalten der Ausgabe

Job Control

^Z	Anhalten des laufenden Programms (suspend)
&	Bewirkt, wenn es hinter einem Kommando steht, daß das Kommando als Hintergrundprozeß gestartet wird.
jobs	Anzeige der aus der aktuellen Shell gestarteten Jobs
bg %1	Läßt Job Nr. 1 im Hintergrund weiterlaufen (background)
fg %1	Holt Job Nr. 1 wieder in den Vordergrund (foreground)
stop %1	Hält Job Nr. 1 an (suspend)
kill %1	Terminiert (beendet) Job Nr. 1

History

history	Gibt eine numerierte Liste der zuletzt eingegebenen Befehle aus.
!komman	Führt diejenige Kommandozeile erneut aus, die mit `komman...` begann.
!10	Führt das Kommando Nr. 10 aus der `history`-Liste erneut aus.
!10 -l /etc	Führt das Kommando Nr. 10 aus der `history`-Liste mit neuen Parametern aus.

Hinweise: Wird die aktuelle Shell beendet, laufen nur die im Hintergrund gestarteten Jobs weiter, die nicht im Zustand „suspended“ sind, also keine Aus- und Eingaben mehr erwarten. Alle anderen, von dieser Shell abhängigen Jobs werden terminiert.

Ein häufiger Fehler ist es, ein Programm ohne die vorgesehenen Argumente (z.B. Dateinamen oder Optionen) zu starten.

Beispiel: `grep -i /etc/passwd`
In diesem Beispiel würde `grep` von der Standardeingabe lesen und alle eingetippten Zeilen, die den String `/etc/passwd` enthalten, ausgeben (was sicher nicht im Sinne des Aufrufers war, der vermutlich eher nach einem Namen in der Datei `/etc/passwd` suchen wollte, und diesen Namen hinter dem `-i` anzugeben vergaß). Beendet werden solche interaktiv arbeitenden Fehlläufer mit `Control-C` oder `Control-D` am Zeilenanfang.

Shell-interne Kommandos (bash)

Beispiel	Bedeutung
`alias ll='ls -l'`	Erzeugt die Abkürzung `ll` für das Kommando `ls -l`
`set`	Gibt aktuelle Einstellungen der Shell aus
`env`	Zeigt vererbbare Umgebungsvariablen an (environment)
`export VARIABLE="Wert"`	Setzt eine vererbbare Variable
`export PATH="$HOME/bin:$PATH"`	Stellt dem Suchpfad das zusätzliche Verzeichnis `Benutzerhomeverzeichnis/bin` voran
`which Kommando`	Zeigt an, unter welchem Verzeichnis im Suchpfad das Kommando installiert ist.

Quotes und Wildcards

`\`	Verhindert das Auswerten des folgenden Zeichens (beispielsweise ein `&`, welches ansonsten Hintergrundprozesse starten würde) durch die Shell.
`" "`	Angabe von Namen, die Leer- oder Sonderzeichen enthalten. Shell-Variablen wie `$PATH` werden aber durch ihren Inhalt ersetzt.
`' '`	Angabe von Namen, die Leer- und Sonderzeichen inclusive `$` enthalten.
`` ` ` ``	Das Kommando innerhalb der Backquotes wird ausgeführt und durch die Kommandoausgabe ersetzt.
`*`	Ersetzt ein oder mehrere beliebige Zeichen.
`?`	Ersetzt genau ein Zeichen.
`*.{tex,daten}`	Paßt auf alle Dateien mit der Endung `.tex` oder `.daten`
`[a-h]`	Ersetzt alle Buchstaben von `a` bis `h`

Trickreiche Beispiele und hilfreiche Programme

`cd -`	wechselt zurück in das Verzeichnis, aus dem man beim letzten `cd`-Aufruf kam.
`rm '*'`	Löscht eine Datei mit dem Namen `*` aus dem aktuellen Verzeichnis.
`ls -l `which passwd` `	Zeigt die Dateiberechtigungen des Programms `passwd` an, welches irgendwo im Pfad liegen kann.
`rm ./-i`	Löscht eine Datei mit dem Namen `-i` im aktuellen Verzeichnis.
`rm -i *`	Löscht alle Dateien im aktuellen Verzeichnis, fragt bei jeder Datei vorher nach.
`echo *`	erzeugt gleiche Ausgabe wie `ls .`, so können Sie das `ls`-Kommando ersetzen, wenn es nach einem Crash fehlt.
`ps aux | grep ^knopper`	Erzeugt eine ausführliche Prozeßliste aller Prozesse, die vom Benutzer `knopper` gestartet wurden.
`top`	Erzeugt eine sich selbst aktualisierende Liste der Prozesse im System.
`w`	Eines der kürzesten Kommandos, zeigt an, welche Benutzer eingeloggt sind und was sie gerade tun.
`cat /proc/{ioports,interrupts}`	Zeigt die Hardware-Adressen aller an das System angeschlossenen Karten und deren Interruptbelegung an.
`free`	Zeigt die Speicherauslastung an (Linux-spezifisch).
`killall -TERM xterm`	Alle laufenden `xterm`s werden terminiert. (Linux-spezifische Kommandosyntax)