

HOWTO zur Shell-Quotierung

(C) 2006 T.Birnthaler/H.Gottschalk <howtos(at)ostc.de>
OSTC GmbH, <http://www.ostc.de>

\$Id: shell-quoting-HOWTO.txt,v 1.7 2009-03-25 09:20:55 tsbirn Exp \$

Dieses Dokument beschreibt die verschiedenen Verfahren zur Quotierung von Shell-Sonderzeichen.

Inhaltsverzeichnis

- 1) Shell-Sonderzeichen
- 2) Quotierung
- 3) Beispiel

1) Shell-Sonderzeichen

Die Shell kennt eine Reihe von Sonderzeichen, die nicht für sich selbst stehen, sondern von ihr speziell interpretiert werden und dann bestimmte Funktionen auslösen (26 Stück!):

```
+-----+
| *  ?  [  ]  =  $  <  >  |  &  \
| "  '  `  ;  (  )  {  }  ~  ^  !  #
|
| <Space>  <Tabulator>  <Return>
|
+-----+
```

2) Quotierung

Sollen diese Zeichen nicht ihre Sonderbedeutung haben, sondern als ganz normaler Text interpretiert werden, so sind sie vor der Shell zu "schützen", der Fachausdruck dafür ist "quotieren" (zitieren). Die Shell kennt 3 verschiedene Möglichkeiten des Schützens von Zeichen, die für unterschiedliche Anwendungsfälle benötigt werden:

Quot.	Bedeutung
'...'	ALLE Sonderzeichen in ... abschalten [single quote, tick]
"..."	ALLE Sonderz. BIS AUF \$, `, \, ! in ... abschalten [double quote]
\Z	GENAU EIN (das folgende) Sonderzeichen Z abschalten [backslash]

Hier eine Übersicht über die von den Quotierungszeichen geschützten (*) bzw. nicht geschützten (-) Sonderzeichen (B=Beendet, CR=Zeilenende, W=wörtlich übernommen, I=Ignoriert):

Quot.	\	\$VAR	*?[]	'...' \$(...)	"	'	CR	!	Whitespace	Rest
"..."	-	-	*	-	B	*	W	-	*	*
'...'	*	*	*	*	*	B	W	-	*	*
\X	*	*	*	*	*	*	I	*	*	*

Sinn der beiden Quotierungs-Varianten "..." und '...' ist es, zu unterscheiden, ob man den WERT einer Variablen oder die AUSGABE eines Kommandos nach dem Einsetzen in eine Kommandozeile NOCHMALS von der Shell interpretieren lassen möchte oder nicht. Beispiel:

```
VAR="*"      # Der Variablen VAR den Text "*" (STERN) zuweisen
echo $VAR    => Alle Dateinamen des akt. Verz. (2x ausgewertet)
echo "$VAR"  => * (1x ausgewertet)
echo '$VAR'  => $VAR (0x ausgewertet)
```

3) Beispiel

Hier ein grosses Beispiel für die Kombination vieler Sonderzeichen in einem Shell-Kommando (statt '...' kann auch \$(...) für die Kommando-Substitution verwendet werden):

```
echo * $TERM `date` > xxx &
  ^ ^      ^ ^ ^ ^ ^
  +-----+-----+-----+-----+-----+-----+-----+
Kommando echo + 6 Argumente, alle
Sonderzeichen werden ausgewertet,
(insbesondere 6x das Leerzeichen)
d.h. alle Dateinamen des akt. Verz.,
der Wert der Variablen TERM und
das aktuelle Datum landen in
```

