Kurzreferenz UNIX-Shellprogrammierung

1

Starten von Skripten	Bedeutung
ksh *shellscript* **shellscript** **. [pfad/]shellscript**	Neue Subshell verarbeitet Skript Skriptaufruf, wenn r-x und $PATH Skript läuft in akt. Shell ! (sourcen)
Start / Initialisierungsdateien	
/etc/profile **~/.profile** **~/.kshrc**	Systemweite Startdatei pers. Init-datei für ksh / bash pers. Init-datei spez. für ksh
Alias	
alias **alias** *name='befehl'* **alias** *-t* **alias** *-x* **alias** *-x name='befehl'* **alias** *-t commando* **unalias** *aliasname*	Zeigt alle aliase an Setzt neuen Alias Anzeige aller tracked aliase Anzeige aller exportierten aliase Setzt exportierten alias Setzt tracked alias Löscht alias
#! /bin/ksh # Kommentartext	Mit welcher Shell soll Skript verarbeitet werden / Info-Text
Kommadoverkettung	
kmd ; kmd (kmd ; kmd) {kmd ; kmd;} kmd **&&** kmd kmd **||** kmd kmd **|** kmd	Sequenzielle Verarbeitung Gruppierung in Subshell Gruppierung ohne Subshell Bedingte Ausführung UND Bedingte Ausführung ODER Pipe: output wird input
Kommadosubstitution	
`*befehl*` **oder $(***befehl***)**	Kommandosubstitution: Die Ausgabe von *befehl* wird die Kmd-Zeile eingesetzt.
Funktionen	
functionname() **{** kommandos **}**	Funktionsdeklaration Übergabe von Parametern analog zu Skript ($1 -$n) ACHTUNG: gleicher Adressraum.

2

Variablen	
var=	Erzeugt Variable
var=wert	Erzeugen und Wert zuweisen
var=$var2	Wert aus var2 zuweisen
var=${var2}string	Stringconcatenation
var=varvar	Variablenconcatenation
print oder **echo $var**	Ausgabe des Variableninhalts
set	Anzeige aller Variablen
env	Anzeige aller exportierten Variablen
unset var	Läschen einer Variablen
export var	Exportieren einer Variablen
integer var	Variable ist integer
readonly var	Variable ist readonly
arrayname[index]=wert	Erzeugt ein Array mit Werten
Interne Variablen und Positionsparameter	
$$	PID der aktuellen Shell
$!	PID letzter Hintergrundprozess
$?	Exit-Status des letzten Kommandos
$0	Shellskript mit absolutem Pfad
$1 $2 … $9	Aufrufparameter 1 – 9 (Argumente)
${10} ${11} … ${n}	Aufrufparameter 10 – n
$@	Alle Parameter 1-n "a1" "a2" …
$*	Alle Parameter 1-n "a1 a2 …"
$#	Anzahl aller Parameter
set --	Löscht aller Parameter
set [--] *parameterliste*	Setzt Parameterliste neu
shift *[n]*	Verschiebt $2 zu $1 …
Variablendefinition über typeset	
typeset –L[n] var=[wert]	var ist linksbündig mit Länge n
typeset –R[n] var=[wert]	var ist linksbündig mit Länge n Füllzeichen = BLANK
typeset –Z[n] var=[wert]	var ist rechtsbündig mit Länge n Füllzeichen = 0
typeset –l var=[wert]	lowercase, Inhalt in *klein* Buchst.
typeset –u var=[wert]	uppercase, Inhalt in *GROSS* Buchst.
typeset –i var=[wert]	var ist ***integer*** , wie **integer** *var*
typeset –r var=[wert]	var ist ***readonly***
typeset –x var=[wert]	wie **export** *var*

3

Variablenverwendung	
${#var}	Länge der Variablen *var*
${var:?"text"}	var 0 oder unset: exit mit Ausg. *text*
${var:-defaultwert}	var 0 oder unset: verwende *default*
${var:=defaultwert}	var 0 oder unset: verwende *default und weise wert der var zu*
${var#muster}	entferne kürzestest muster von →
${var##muster}	entferne längstes muster von var →
${var%muster}	entferne kürzestest muster von var ←
${var%%muster}	entferne längstes muster von var ←
${array[index]}	Inhalt von array an position *index*
${#array[*]}	Anzahl belegter Felder in array
${array[*]}	Alle Elemente aus array
Schleifen	
for var **in** element1 element2 … **do** kommandos **done**	Elementenschleiife über alle Elemente. Schleifenende, wenn kein Element mehr in Liste
while kommando **do** kommandos **done**	Schleife mit Bedingung. Schleifenende, wenn Exitstatus von *kommando* ($?) false / ungleich 0
until *kommando* **do** *kommandos* **done**	Schleife mit Bedingung. Schleifenende, wenn Exitstatus von *kommando* ($?) true / gleich 0
Schleifen Sprünge und Abbruch	
continue	Sprung an den Schleifenkopf
break	Sprung an Schleifenende / Abbruch der Schleife
Script Abbruch	
exit	Normaler Ausstieg
exit 0	Ausstieg mit $? = 0 (Erfolg)
exit 1 - 255	Ausstieg mit $? > 0 (Fehler)

Kurzreferenz UNIX-Shellprogrammierung

4

Verzweigungen	
if testkommando **then** kommando **else** kommando **fi**	**wenn** *testkommando* = true **dann** kommandoliste ... **sonst** kommandoliste ... **ende**
if testkommando **then** kommandos **elif** testkommando kommandos **else** kommandos **fi**	**wenn** *testkommando* = true **dann** kommandoliste ... **sonst-wenn** *testkommando* = true kommandoliste ... **sonst** kommandoliste ... **ende**
Mehrfache Verzeigung	
case wort **in** **muster1)** kommandos ;; **muster2)** kommandos ;; *) kommandos ;; **esac**	**wenn** wort **passt auf** **muster1** , dann kommando und ende **muster1** , dann kommando und ende Default Fall **ende** *wort* kann $var sein
Metazeichen	
*	ein, kein oder mehrere Zeichen
?	genau ein Zeichen
[ ]	ein Zeichen aus angegebener Menge
[! . .]	ein Zeichen **nicht** aus ang. Menge
\	maskiert folgendes Sonderzeichen
'...'	maskiert **alle** Sonderzeichen im eingeschlossenen Text
"..."	Maskierung der Sonderzeichen im eingeschlossenen Text mit Ausnahme von $ \ `...`
kommando **&**	Ausführung im Hintergrund

5

Text Ein- und Ausgabe	
read [var [var1 var2]]	
-un	Lesen vom Kanal n (mit exec öffnen)
-r	raw mode. Zeilenende kann nicht mit \ maskiert werden
-p	Lesen aus einer Pipe (Hintergrund-prozess)
print	
--	nächstes Argument ist keine Option
-n	kein Newline (identisch mit \c)
-un	Ausgabe nach Kanal n (eventuell mit exec öffnen)
-r oder –R	raw Modus, Escape-Sequenzen ignorieren
-p	Ausgabe auf Pipeline (Hintergrund-prozess)
-s	Ausgabe in History-Datei
Escape-Sequenz für print und echo -e	
\a	Systempiep (Alarm)
\b	Backspace (CTRL-H)
\c	Newline am Zeilenende unterdrücken
\f	Formfeed (Seitenvorschub)
\n	Newline (CTRL-J)
\r	Return (CTRL-M)
\t	Tabulator (CTRL-I)
\v	Vertikaler Tabulator (CTRL-K)
\0n	ASCII-Zeichen in oktalem Wert n
****	ein Backslash

6

Kanalumlenkung	
kdo n> datei	Kanal n nach datei umlenken
kdo n>> datei	Kanal n an datei anhängen
kdo n< datei	Kanal n liest von datei
kdo < datei	stdin von datei nehmen
kdo1 | kdo2	Pipe: stdout von kdo1 als stdin von kdo2 verwenden
kdo >| datei	stdout auf datei zwingen, auch wenn die Option noclobber aktiv ist
kdo <> datei	datei für Ein- und Ausgabe verwenden
kdo << [-]marke **...** **marke**	Here-document: stdin von kdo wird aus Folgezeilen genommen (bis marke)
kdo m>&n	Umlenken von Kanal m nach Kanal n
kdo m<&n	Kanal m liest aus derselben Quelle wie Kanal n
Filediskriptoren	**Zusätzliche Kanäle 3 bis 9**
exec n>datei	Öffnet Kanal n schreibend auf Datei
exec n>>datei	Öffnet Kanal n anhängend auf Datei
exec n<datei	Öffnet Kanal n lesend auf Datei
exec n<>datei	Kanal n wird mit datei zum Lesen und Schreiben verbunden
exec kdo	Aktuelle Shell wird durch kdo überlagert
exec n<&- / **exec n>&-**	Kanal n wieder schließen
Berechnung / Arithmetischer Vergleich	
((Vergleich / Berechnug **))**	Rückgabe nur Exitstatus in $?
$((Vergleich / Berechnug **))**	Ergebnis oder Exitstaus in Kmd-Zeile

Kurzreferenz UNIX-Shellprogrammierung

7

Vergleichsoperator	Bedeutung
<	kleiner
>	größer
<=	kleiner gleich
>=	größer gleich
==	gleich
!=	ungleich
Verknüpfung	**Bedeutung**
(...)	Klammerung
&&	logisches Und
||	logisches Oder
Rechenoperator	**Bedeutung**
+	Addition
-	Subtraktion
*	Multiplikation
/	Division
%	Modulo (Rest der Division)
<<	Bit-shift links
>>	Bit-shift rechts
&	Bit-weises Und
|	Bit-weises Oder
~	Bit-weise Negation
^	Bit-weises exklusives Oder

8

test, [...], [[...]]	Dateiattribute
-a dat oder –e dat	dat existiert (beliebiger Typ)
-f dat	dat ist eine reguläre Datei (file)
-d dat	dat ist ein Directory
-L dat	dat ist ein symbolischer Link
-b dat	dat ist eine blockorientierte Gerätedatei
-c dat	dat ist eine character-orientierte Gerätedatei
-r dat	dat ist lesbar (read)
-w dat	dat ist schreibbar (write)
-x dat	dat ist ausführbar (execute)
-s dat	dat existiert und ist nicht leer (size)
-O dat	gleiche UID wie Eigentümer von dat (owner)
-G dat	gleiche GID wie die Gruppe
dat1 –nt dat2	dat1 neuer als dat2 (newer than)
dat1 –ot dat2	dat1 älter als dat2 (older than)
dat1 –ef dat2	dat1 und dat2 sind identisch (equal file: hardlink)
test, [...], [[...]]	**Integervergleiche**
z1 –eq z2	gleich (equal)
z1 –ne z2	ungleich (not equal)
z1 –lt z2	kleiner (less than)
z1 –le z2	kleiner gleich (less equal)
z1 –gt z2	größer (greater than)
z1 –ge z2	größer gleich (greater equal)

9

test bzw. [...]	Stringvergleiche	[[...]]
str = string	str wird durch muster beschrieben (Mustervergleich)	str = muster
str != string	str wird nicht durch muster beschrieben	str != muster
-n str	str ist nicht leer (non zero)	-n str
-z str	str ist leere Zeichenkette (zero)	-z str
	str1 ascii-mäßig kleiner als str2	str1 < str2
	str1 ascii-mäßig größer als str2	str1 > str2
test bzw. [...]	Verknüpfungen	[[...]]
\(... \)	Gruppieren	(...)
! bed	Negation	! bed
bed1 –a bed2	logisches UND	bed1 && bed2
bed1 –o bed2	logisches ODER	bed1 || bed2
crontab	Wiederholte Ausführung	

-e pers. crontab editieren
-l pers. crontab anzeigen
-r pers. crontab löschen

Minute	Stunde	Tag(Monat)	Monat	Tag(0=7=So)	Kommando
35	7	1	*	*	rm –rf /tmp/...

© Integrata AG
© Integrata AG

Kurzreferenz UNIX-Shellprogrammierung

10

Optionen für ksh und bash	
set –o *option(en)*	Korn-/Bourne-Again-Shell-Optionen setzen
set +o *option(en)*	Aufheben von Optionen
Option	Beschreibung
allexport	Neu definierte Shellvariablen werden automatisch zu Umgebungsvariablen.
bgnice	Hintergrundprogramme erhalten eine niedrigere Prozesspriorität als die aufrufenden Shell.
emacs	Die Befehlszeile kann mit den Funktionen des Editors emacs bearbeitet werden.
errexit	Die Shell wird bei Auftreten eines Fehlers beendet.
gmacs	Die Befehlszeile kann mit den Funktionen des Editors gmacs bearbeitet werden, der GNU-Version von emacs.
ignoreeof	Es nicht mehr möglich, eine Shell durch die Tastenkombination Control-d oder Strg-d zu beenden.
interactive	Alle dem Hashzeichen # folgenden Zeichen gelten als Kommentare.
keyword	Alle Variablen eines Befehls werden automatisch zu Shellvariablen.
markdirs	Alle Verzeichnisse erhalten bei der Ausgabe einen nachfolgenden Schrägstrich /, wenn der Befehl ein Metazeichen enthält, wie zum Beispiel ls *.
monitor	Wenn Hintergrundprozesse enden, werden Meldungen ausgegeben.
noexec	Befehle werden nicht ausgeführt, sondern nur ihre Richtigkeit geprüft.

11

noclobber	Bei der Ausgabeumlenkung werden bereits bestehende Dateien nicht überschrieben.
noglob	Die Shell kann keine Dateinamen mit Hilfe von Sonderzeichen erstellen.
nolog	In der History-Liste werden keine Funktionen gespeichert.
notify	Es wird das Ende eines Hintergrundprozesses angezeigt, ohne dass die Return-Taste gedrückt wird.
nounset	Abbruch eines Befehls, wenn er nicht vorhandene Variablen verwendet.
privileged	Die Initialisierungsdateien (.profile und .kshrc) des Benutzers werden nicht gelesen, sondern die Datei /etc/suid_profile.
restricted	Der Benutzer wird bei der Bedienung der Shell stark eingeschränkt (Restricted Shell).
trackall	Jeder Befehl wird beim ersten Aufruf mit seinem Pfad als Alias angelegt und nicht mehr über die Variable `PATH` gesucht (tracked alias).
verbose	Anzeige eines Befehls vor der Ausführung, so wie er von der Shell gelesen wird.
vi	Die Befehlszeile kann mit den Funktionen des Editors vi bearbeitet werden.
viraw	Jedes Zeichen wird wie im Editor vi bearbeitet.
xtrace	Anzeige eines Befehls vor der Ausführung, so wie er von der Shell interpretiert wird.

12

awk	**A**ho , **W**einberger, **K**ernighan
awk ' BEGIN { action } pattern { action } END { action } ' inputfile	awk liest aus inputfile
kmd | **awk** ' { } '	awk liest aus pipe
awk Variablen	
$0	Inhalt ist aktueller Record / akt. Zeile
$1 - $100	Inhalte sind Felder/Wote von akt. Zeile
NF	Anzahl Felder / Worte
NR	Zeilen- / Recordnummer (alle Dateien)
FNR	Zeilen- / Recordnummer aktuelle Datei
FILENAME	aktueller Dateiname
FS	Eingabe Trennzeichen
OFS	Ausgabe Trennzeichen
RS	Eingabe Zeilentrenner
ORS	Ausgabe Zeilentrenner
sed	**S**tream**ed**itor
sed ' **kmd** ' inputfile	
trap	
trap '' SIGNAL	Signal wird ignoriert
trap ' kmd ' **SIGNAL**	Signal löst kmd aus (SIG umdefiniert)
trap SIGNAL	Signal auf default zurück setzen
trap	umdefinierte Signale anzeigen
tar	**t**ape **Ar**chivierung
-c	Erzeugt tar-Archiv
-p	Erhält Rechte / Owner
-f datei.tar	Angabe tar-Archiv
-x	Entpackt tar-Archiv
-z	Komprimiert mit zip / gzip