FORSCHUNGSZENTRUM JÜLICH GmbH KFA

Zentralinstitut für Angewandte Mathematik
D-52425 Jülich, Tel. (02461) 61–6402
Informationszentrum, Tel. (02461) 61–6658

Referenzkarte

KFA-ZAM-RFK-0011

U. Schmidt

15.11.94

Korn Shell Programming

Note: In general brackets are used to indicate that the information can be omitted. If brackets are part of the clause they are marked by *.

Invocation and initilization

(1) **ksh [*options*] [*filename*] [*args*]**
A new shell is created and if *filename* is specified, the commands in *filename* will be executed.

(2) ***filename* [*args*]**
A new shell is created and the commands in *filename* are executed.

(3) **. *filename***
The commands in *filename* are executed within the current shell.

Initialization files:

/etc/profile	System default shell startup. Excecuted only at login time.
$HOME/.profile	User's shell startup. Executed only at login time.
$ENV	Name of ksh startup file. Executed each time a new shell is created. Normally the name of a file, which then calls **$HOME/.kshrc**

Command syntax

#	Following text on the line is comment
\	Use \ as last character to indicate that a continuation line is following
cmd1; cmd2	Run cmd1, then run cmd2
cmd1 && cmd2	Run cmd1, then run cmd2 only, if cmd1 succeeded
cmd1* || *cmd2	Run cmd1, then run cmd2 only, if cmd1 failed
***cmd1*	*cmd2***	Define a pipeline

User defined shell variables

Simple variables:

Assignment: ***NAME=value***
Reference: ***$NAME***

Array variables:

Assignment: ***NAME[index]=value****
Reference: ***$NAME[index]****

Commands:

To use variables in subsequently called shells, they must be *exported*.

export *NAME[=value]*	Add variable *NAME* to export list
env	List all exported variables

Special $ variables

$0	Name of command/script currently executed
$*n* or **${*nn*}**	Value of parameter at position *n*; for *n* greater 9, enclose *nn* in braces
$*	String with all positional parameters
$#	Number of positional parameters
$?	Exit status of last executed command
$$	Process ID of actual shell. Used to generate unique file names.

Shell environment variables used by Korn shell

HOME	User's home directory
PATH	List of names of directories to search for executable commands
PS1	User's initial prompt string
PS3	Prompt string for select; default '#?'
PS4	Prompt string for set -xv; default '+'
USER	User's login name
TERM	terminal type
DISPLAY	X11 server display

Shell environment variables set by Korn shell

LINENO	Line number of the current script
OPTARG	Value of last option processed by getopts
OPTIND	Index of last option processed by getopts
PPID	Process ID of the parent shell
PWD	Current working directory
RANDOM	A random number between 0 and 32767
REPLY	Set by the select command

Shell Parameter substitution

${#*name*}
${#*name*[*index*]}*
Length of the value of variable *name* or $*n*, or array element with index *index*

${#*name*[*]}*
Number of defined elements in the array *name*

${*name*:=*value*}
If *name* isn't null, then it is used; otherwise *value* is used and assigned to *name*.

${*name*:-*value*}
If *name* isn't null, then it is used; otherwise *value* is used but not assigned to *name*.

${*name*:+*value*}
If *name* is null, then null is used; otherwise *value* is used but not assigned to *name*.

${*name*#*value*}
${*name*##*value*}
Search for pattern is done from left to right (beginning) and the rightmost string without pattern is substituted. The 1. form splits at the 1., the 2. form at the last ocurrence.

${*name*%*value*}
${*name*%%*value*}
Search for pattern is done from right to left (ending) and the leftmost string without pattern is substituted. The 1. form splits at the 1., the 2. form at the last ocurrence.

Shell arithmetic

(1) **expr *expression***
***name*=`expr *expression*`**
Bourne compatible. Each element of *expression* must be separated by blanks.

(2) **typeset -i *name***
name=expression
The elements of *expression* must be coded as one string without blanks.

Expression is a combination of *terms* and *operators*, with or without parenthesis. Each *term* may be an integer variable or integer constant.

term op term [***op term*** [...]]

arithmetic *operators* are: ***** **/** **%** **+** **–** **<<** **>>**
comparison *operators* are: **<** **<=** **>** **>=** **==** **!=**
logical *operators* are: **&** **^** **|** **&&** **||**

Notice that some operators must be quoted or backslashed to avoid confusion with the wild card characters or shell symbols for I/O redirection.

Conditional expressions

... for files; true if

-a ***file***	*file* exists
–d ***file***	*file* exists and is a directory
-f ***file***	*file* exists and is an ordinary file
-r ***file***	*file* exists and is readable
-s ***file***	*file* exists and has a size greater than 0
–w ***file***	*file* exits and is writable
-x ***file***	*file* exists and is executable
-L ***file***	*file* exists and is a symbolic link
-O ***file***	*file* exists and owned by user
file1 **-nt** ***file2***	*file1* exists and is newer than *file2*
file1 **-ot** ***file2***	*file1* exists and is older then *file2*

... for strings: true if

–n ***str***	*string* length is greater than 0
-z ***str***	*string* length is zero
str1 **=** ***str2***	*string1* matches *string2*
str1 **!=** ***str2***	*string1* does not match *string2*
str1 **<** ***str2***	*string1* comes before *string2* (ASCII)
str1 **>** ***str2***	*string1* comes after *string2* (ASCII)

... for integer values: true if

n1 **-eq** ***n2***	*n1* is equal to *n2*
n1 **-ne** ***n2***	*n1* is not equal to *n2*
n1 **-lt** ***n2***	*n1* is less than *n2*
n1 **-le** ***n2***	*n1* is less than or equal to *n2*
n1 **-gt** ***n2***	*n1* is greater than *n2*
n1 **-ge** ***n2***	*n1* is greater than or equal to *n2*

Wild card characters and pattern

*****	Any string, including nullstring
?	Any single character
[abc]	Any of the enclosed characters abc
[a-z]	Any of the enclosed characters in the range a through z

Control statements

... for conditional expressions

test ***expression*** or **[** ***expression*** **]***

expression may be a compound expression with following operators: **!** (NOT), **-a** (AND) or **-o** (OR).

[[***expression*** **]]***

expression may be a compound expression with following operators: **!** (NOT), **&&** (AND) or || (OR).

if ***list1*****; then** ***list2*****; [else** ***list3*****;] fi**

if ***list1*****; then** ***list2*****; elif** ***list3*****; [elif** ***list4*****; [...]] [else** ***listn*****;] fi**

case ***word*** **in [[(] *pattern1* [| *pattern2*]) *list* ;;] [...] esac**

select ***var*** **[in** ***words*** **]; do** ***list*****; done**

... for loops

for ***var*** **[in** ***words*** **]; do** ***list*****; done**

while ***list1*****; do** ***list2*****; done**

until ***list1*****; do** ***list2*****; done**

... for functions

function_identifier**( ) { *list*; }**

Command line editing (History)

<ESC> k [k] [j]	Scrolling within shell history to retrieve commands. **k**: fetch previous command, up; **j**: fetch next command, down. If there are problems on terminal emulation within the net, use **set -o vi**.

Command mode (activated by <ESC>)

h	move cursor back, left, one character
l	move cursor forward, right, one character
w	move cursor to begin of next word
b	move cursor to begin of previous word
x	delete character at cursor
D	delete rest of line starting at cursor

Input mode

i | **>**	insert characters before | after cursor
r	replace one character at cursor
R	replace characters starting at cursor
A	append charcters at end of line

for more information see vi-RFK-0010

Builtin commands

:	Nullstatement
alias [-tx] [*name*[=*value*]]	List or define an abbreviation
break [*n*]	Escape from loops or case
cd [*dirname*]	Change the current directory
continue [*n*]	Start next iteration of a loop
echo [*arg*]	Write arguments to standard output
eval [*arg*]	Evaluate arguments
exit [*n*]	Exit from current shell
getopts *optstr name* [*arg*]	Parse command line options and arguments
let *expression* or **((*expression*))**	Evaluate an arithmetic expression
pwd	Print working directory
read [*arg*]	Read a line from standard input
readonly [*name*]	Prevent alteration of selected variables
return [*n*]	Return from a shell function
set [*options*] [arg]	Set shell options and $*n* variables by arguments; 'set -xv' enables tracing and uses PS4
shift [*n*]	Shift arguments left
test *expression* or **[*expression*]***	Evaluate a conditional expression
trap [*omd*] [*signal* ...]	Specify exceptional condition handling
type [*name*]	Identify a command
typeset *options* [*name*=[*value*]]	Define characteristics of a variable; *options* may be **-i** for integer, **-l** for lowercase, **-u** for uppercase, **-L*n*** for left substring, **-R*n*** for right substring
unalias *name*	Drop an alias
umask [*nnn*]	Set or display the file creation mask
unset *name*	Drop a shell variable

Information at KFA

Information Centre / Dispatch:

Phone: +49–2461–61–6658 or 5642
Telefax: +49–2461–61–2810
E-Mail: zam.beratung@kfa-juelich.de