# Korn Shell编程(Unix AIX)

当命令是通过文件来执行，该文件成为shell脚本，并且shell以非交互式运行。

第一行：置于顶端第一行，内核根据它来确定该用哪个程序来翻译脚本中的行。

﻿#!/bin/ksh

变量?：通过$?方式来访问，包含了最后被执行的命令的退出状态。0表示命令成功退出，非0表示因为某种原因导致失败。

使用双方括号字符串表达式和文件的测试，使用双圆括号(C语言的风格)的数字操作符来测试数字表达式。

数字：

if(($# < 1))

then

print “$0:arguments “ 1>&2

exit 1

fi

字符串：

print “Are you ok (y/n)?”

read answer

if[[ “$answer” = [Yy]\* ]

then

print “ok”

fi

## tar

**使用方式**

tar -{c|r|t|u|x} [ -BdDEFhilmopRUsvwZ ] [ -Number ] [ -f TarFil e ]

[ -b Blocks ] [ -S [ Feet ] | [ Feet@Density ] | [ Blocksb ] ]

[ -L InputList ] [-X ExcludeFile] [ -N Blocks ] [ -C Directory ] File..

**参数**

-c ：建立一个压缩文件的参数指令(create 的意思)；

-x ：解开一个压缩文件的参数指令！

-t ：查看 tarfile 里面的文件！

特别注意，在参数的下达中， c/x/t 仅能存在一个，不可同时存在！

因为不可能同时压缩与解压缩。

-z ：是否同时具有 gzip 的属性？亦即是否需要用 gzip 压缩？

-j ：是否同时具有 bzip2 的属性？亦即是否需要用 bzip2 压缩？

-v ：压缩的过程中显示文件！这个常用，但不建议用在背景执行过程！

-f ：使用档名，请留意，在 f 之后要立即接档名喔！不要再加参数！

　　　例如使用『 tar -zcvfP tfile sfile』就是错误的写法，要写成

　　　『 tar -zcvPf tfile sfile』才对喔！

-p ：使用原文件的原来属性（属性不会依据使用者而变）

-P ：可以使用绝对路径来压缩！

-N ：比后面接的日期(yyyy/mm/dd)还要新的才会被打包进新建的文件中！

**范例**

tar –cvf test.tar test 将test文件打包成test.tar，相对路径

tar –xvf test.tar 解压缩test.tar

tar -zcvf /tmp/etc.tar.gz /etc打包后，以 gzip 压缩

## scp

# Shell 命令(Linux)

## cat

使用权限：所有使用者   
　　使用方式：cat [-AbeEnstTuv] [--help] [--version] fileName   
　　说明：把档案串连接后传到基本输出（萤幕或加 > fileName 到另一个档案）   
　　参数：   
　　-n 或 --number 由 1 开始对所有输出的行数编号   
　　-b 或 --number-nonblank 和 -n 相似,只不过对于空白行不编号   
　　-s 或 --squeeze-blank 当遇到有连续两行以上的空白行,就代换为一行的空白行   
　　-v 或 --show-nonprinting   
　　范例：   
　　cat -n textfile1 > textfile2 把 textfile1 的档案内容加上行号后输入 textfile2 这个档案里   
　　cat -b textfile1 textfile2 >> textfile3 把 textfile1 和 textfile2 的档案内容加上行号（空白行不加）之后将内容附加到 textfile3

## cd

使用权限:所有使用者   
　　使用方式:cd [dirName]  
　　说明:变换工作目录至 dirName。 其中 dirName 表示法可为绝对路径或相对路径。若目录名称省略,则变换至使用者的 home directory (也就是刚 login 时所在的目录).另外,"~" 也表示为 home directory 的意思,"." 则是表示目前所在的目录,".." 则表示目前目录位置的上一层目录。   
　　范例:跳到 /usr/bin/:  
　　cd /usr/bin   
  
　　跳到自己的 home directory:  
　　cd ~   
  
　　跳到目前目录的上上两层:  
　　cd ../..

## chmod

使用权限:所有使用者   
　　使用方式:chmod [-cfvR] [--help] [--version] mode file...   
　　说明:Linux/Unix 的档案存取权限分为三级:档案拥有者,群组,其他。利用 chmod 可以藉以控制档案如何被他人所存取。  
　　把计:  
　　mode:权限设定字串,格式如下:[ugoa...][[+-=][rwxX]...][,...],其中u 表示该档案的拥有者,g 表示与该档案的拥有者属于同一个群体(group)者,o 表示其他以外的人,a 表示这三者皆是。   
　　+ 表示增加权限,- 表示取消权限,= 表示唯一设定权限。   
　　r 表示可读取,w 表示可写入,x 表示可执行,X 表示只有当该档案是个子目录或者该档案已经被设定过为可执行。   
　　-c:若该档案权限确实已经更改,才显示其更改动作   
　　-f:若该档案权限无法被更改也不要显示错误讯息   
　　-v:显示权限变更的详细资料   
　　-R:对目前目录下的所有档案与子目录进行相同的权限变更(即以递回的方式逐个变更)   
　　--help:显示辅助说明   
　　--version:显示版本  
　　范例 :将档案 file1.txt 设为所有人皆可读取:  
　　chmod ugo+r file1.txt   
  
　　将档案 file1.txt 设为所有人皆可读取:  
　　chmod a+r file1.txt   
  
　　将档案 file1.txt 与 file2.txt 设为该档案拥有者,与其所属同一个群体者可写入,但其他以外的人则不可写入:  
　　chmod ug+w,o-w file1.txt file2.txt   
  
　　将 ex1.py 设定为只有该档案拥有者可以执行:  
　　chmod u+x ex1.py   
  
　　将目前目录下的所有档案与子目录皆设为任何人可读取:  
　　chmod -R a+r \*   
  
　　此外chmod也可以用数字来表示权限如 chmod 777 file   
　　语法为：chmod abc file   
　　其中a,b,c各为一个数字,分别表示User,Group,及Other的权限。  
  
　　r=4,w=2,x=1   
　　若要rwx属性则4+2+1=7；   
　　若要rw-属性则4+2=6；   
　　若要r-x属性则4+1=7。   
  
　　范例：   
　　chmod a=rwx file   
　　和   
　　chmod 777 file   
　　效果相同   
　　chmod ug=rwx,o=x file   
　　和   
　　chmod 771 file   
　　效果相同   
　　若用chmod 4755 filename可使此程式具有root的权限

## chown

使用权限:root   
　　使用方式:chown [-cfhvR] [--help] [--version] user[:group] file...   
　　说明:Linux/Unix 是多人多工作业系统,所有的档案皆有拥有者。利用 chown 可以将档案的拥有者加以改变。一般来说,这个指令只有是由系统管理者(root)所使用,一般使用者没有权限可以改变别人的档案拥有者,也没有权限可以自己的档案拥有者改设为别人。只有系统管理者(root)才有这样的权限。   
　　把计:  
　　user:新的档案拥有者的使用者

IDgroup:新的档案拥有者的使用者群体(group)

-c:若该档案拥有者确实已经更改,才显示其更改动作

-f:若该档案拥有者无法被更改也不要显示错误讯息

-h:只对于连结(link)进行变更,而非该 link 真正指向的档案

-v:显示拥有者变更的详细资料

-R:对目前目录下的所有档案与子目录进行相同的拥有者变更(即以递回的方式逐个变更)

--help:显示辅助说明

--version:显示版本   
　　范例:  
　　将档案 file1.txt 的拥有者设为 users 群体的使用者 jessie:  
　　chown jessie:users file1.txt   
  
　　将目前目录下的所有档案与子目录的拥有者皆设为 users 群体的使用者 lamport:  
　　chmod -R lamport:users \*

## cp

使用权限：所有使用者   
　　使用方式：   
　　cp [options] source dest   
　　cp [options] source... directory   
　　说明：将一个档案拷贝至另一档案,或将数个档案拷贝至另一目录。  
　　把计:   
　　-a 尽可能将档案状态,权限等资料都照原状予以复制。   
　　-r 若 source 中含有目录名,则将目录下之档案亦皆依序拷贝至目的地。   
　　-f 若目的地已经有相同档名的档案存在,则在复制前先予以删除再行复制。   
　　范例：   
　　将档案 aaa 复制(已存在),并命名为 bbb:  
　　cp aaa bbb   
  
　　将所有的C语言程式拷贝至 Finished 子目录中:  
　　cp \*.c Finished

## cut

使用权限：所有使用者   
　　用法：cut -cnum1-num2 filename   
　　说明：显示每行从开头算起 num1 到 num2 的文字。   
　　范例：   
　　shell>> cat example   
　　test2   
　　this is test1   
　　shell>> cut -c0-6 example ## print 开头算起前 6 个字元   
　　test2   
　　this i

## find

用法:find [-h|-l] path –list [expression]

使用说明:  
　　将档案系统内符合 expression 的档案列出来。你可以指要档案的名称,类别,时间,大小,权限等不同资讯的组合,只有完全相符的才会被列出来。   
　　find 根据下列规则判断 path 和 expression,在命令列上第一个 - ( ) , ! 之前的部份为 path,之后的是 expression。如果 path 是空字串则使用目前路径,如果 expression 是空字串则使用 -print 为预设 expression   
　　expression 中可使用的选项有二三十个之多,在此只介绍最常用的部份。   
　　-mount, -xdev:只检查和指定目录在同一个档案系统下的档案,避免列出其它档案系统中的档案   
　　-amin n:在过去 n 分钟内被读取过   
　　-anewer file:比档案 file 更晚被读取过的档案   
　　-atime n:在过去 n 天过读取过的档案   
　　-cmin n:在过去 n 分钟内被修改过   
　　-cnewer file :比档案 file 更新的档案   
　　-ctime n:在过去 n 天过修改过的档案   
　　-empty:空的档案-gid n or -group name:gid 是 n 或是 group 名称是 name   
　　-ipath p, -path p:路径名称符合 p 的档案,ipath 会忽略大小写   
　　-name name, -iname name:档案名称符合 name 的档案。iname 会忽略大小写   
　　-size n:档案大小 是 n 单位,b 代表 512 位元组的区块,c 表示字元数,k 表示 kilo bytes,w 是二个位元组。-type c:档案类型是 c 的档案。   
　　d: 目录   
　　c: 字型装置档案   
　　b: 区块装置档案   
　　p: 具名贮列   
　　f: 一般档案   
　　l: 符号连结   
　　s: socket   
　　-pid n:process id 是 n 的档案   
  
　　你可以使用 ( ) 将运算式分隔,并使用下列运算。   
　　exp1 -and exp2   
　　! expr   
　　-not expr   
　　exp1 -or exp2   
　　exp1, exp2   
　　范例:   
　　将目前目录及其子目录下所有延伸档名是 c 的档案列出来。   
　　# find . -name "\*.c"   
　　将目前目录其其下子目录中所有一般档案列出   
　　# find . -ftype f   
　　将目前目录及其子目录下所有最近 20 分钟内更新过的档案列出   
　　# find . -ctime -20

## less

使用权限：所有使用者   
　　使用方式：   
　　less [Option] filename   
　　说明：   
　　less 的作用与 more 十分相似,都可以用来浏览文字档案的内容,不同的是 less 允许使用者往回卷动   
　　以浏览已经看过的部份,同时因为 less 并未在一开始就读入整个档案,因此在遇上大型档案的开启时,会比一般的文书编辑器(如 vi)来的快速。

## ln

使用权限:所有使用者   
　　使用方式:ln [options] source dist,其中 option 的格式为:  
　　[-bdfinsvF] [-S backup-suffix] [-V {numbered,existing,simple}]   
　　[--help] [--version] [--]   
  
　　说明:Linux/Unix 档案系统中,有所谓的连结(link),我们可以将其视为档案的别名,而连结又可分为两种:硬连结(hard link)与软连结(symbolic link),硬连结的意思是一个档案可以有多个名称,而软连结的方式则是产生一个特殊的档案,该档案的内容是指向另一个档案的位置。硬连结是存在同一个档案系统中,而软连结却可以跨越不同的档案系统。   
  
　　ln source dist 是产生一个连结(dist)到 source,至于使用硬连结或软链结则由参数决定。   
  
　　不论是硬连结或软链结都不会将原本的档案复制一份,只会占用非常少量的磁碟空间。  
  
　　-f:链结时先将与 dist 同档名的档案删除-d:允许系统管理者硬链结自己的目录-i:在删除与 dist 同档名的档案时先进行询问-n:在进行软连结时,将 dist 视为一般的档案-s:进行软链结(symbolic link)-v:在连结之前显示其档名-b:将在链结时会被覆写或删除的档案进行备份-S SUFFIX:将备份的档案都加上 SUFFIX 的字尾-V METHOD:指定备份的方式--help:显示辅助说明--version:显示版本   
　　范例:  
　　将档案 yy 产生一个 symbolic link:zz   
　　ln -s yy zz   
　　将档案 yy 产生一个 hard link:zz   
　　ln yy xx

## locate

使用权限：所有使用者   
　　使用方式： locate [-q] [-d ] [--database=]   
　　locate [-r ] [--regexp=]   
　　locate [-qv] [-o ] [--output=]   
　　locate [-e ] [-f ] <[-l ] [-c]   
　　<[-U ] [-u]>   
　　locate [-Vh] [--version] [--help]   
　　说明：   
　　locate 让使用者可以很快速的搜寻档案系统内是否有指定的档案。其方法是先建立一个包括系统内所有档案名称及路径的资料库,之后当寻找时就只需查询这个资料库,而不必实际深入档案系统之中了。在一般的 distribution 之中,资料库的建立都被放在 contab 中自动执行。  
　　一般使用者在使用时只要用   
　　# locate your\_file\_name   
　　的形式就可以了。 参数：   
　　-u   
　　-U   
　　建立资料库,-u 会由根目录开始,-U 则可以指定开始的位置。  
　　-e   
  
　　将排除在寻找的范围之外。  
　　-l   
　　如果 是 1．则启动安全模式。在安全模式下,使用者不会看到权限无法看到的档案。这会始速度减慢,因为 locate 必须至实际的档案系统中取得档案的权限资料。   
　　-f   
　　将特定的档案系统排除在外,例如我们没有到理要把 proc 档案系统中的档案放在资料库中。   
　　-q   
　　安静模式,不会显示任何错误讯息。  
　　-n   
　　至多显示 个输出。   
　　-r   
　　使用正规运算式 做寻找的条件。   
　　-o   
　　指定资料库存的名称。   
　　-d   
　　指定资料库的路径   
　　-h   
　　显示辅助讯息   
　　-v   
　　显示更多的讯息   
　　-V   
　　显示程式的版本讯息 范例：   
  
　　locate chdrv:寻找所有叫 chdrv 的档案   
　　locate -n 100 a.out:寻找所有叫 a.out 的档案,但最多只显示 100 个   
　　locate -u:建立资料库

## ls

使用权限:所有使用者   
　　使用方式:ls [-alrtAFR] [name...]   
　　说明:显示指定工作目录下之内容（列出目前工作目录所含之档案及子目录)。  
　　-a 显示所有档案及目录 (ls内定将档案名或目录名称开头为"."的视为隐藏档,不会列出)   
　　-l 除档案名称外,亦将档案型态,权限,拥有者,档案大小等资讯详细列出   
　　-r 将档案以相反次序显示(原定依英文字母次序)   
　　-t 将档案依建立时间之先后次序列出   
　　-A 同 -a ,但不列出 "." (目前目录) 及 ".." (父目录)   
　　-F 在列出的档案名称后加一符号；例如可执行档则加 "\*", 目录则加 "/"   
　　-R 若目录下有档案,则以下之档案亦皆依序列出   
　　范例：   
　　列出目前工作目录下所有名称是 s 开头的档案,愈新的排愈后面:  
　　ls -ltr s\*   
　　将 /bin 目录以下所有目录及档案详细资料列出:  
　　ls -lR /bin   
　　列出目前工作目录下所有档案及目录；目录于名称后加 "/", 可执行档于名称后加 "\*":  
　　ls -AF

## more

使用权限：所有使用者   
　　使用方式：more [-dlfpcsu] [-num] [+/pattern] [+linenum] [fileNames..]   
　　说明：类似 cat ,不过会以一页一页的显示方便使用者逐页阅读,而最基本的指令就是按空白键（space）就往下一页显示,按 b 键就会往回（back）一页显示,而且还有搜寻字串的功能（与 vi 相似）,使用中的说明文件,请按 h 。   
　　参数：-num 一次显示的行数   
　　-d 提示使用者,在画面下方显示 [Press space to continue, q to quit.] ,如果使用者按错键,则会显示 [Press h for instructions.] 而不是 哔 声   
　　-l 取消遇见特殊字元 ^L（送纸字元）时会暂停的功能   
　　-f 计算行数时,以实际上的行数,而非自动换行过后的行数（有些单行字数太长的会被扩展为两行或两行以上）   
　　-p 不以卷动的方式显示每一页,而是先清除萤幕后再显示内容   
　　-c 跟 -p 相似,不同的是先显示内容再清除其他旧资料   
　　-s 当遇到有连续两行以上的空白行,就代换为一行的空白行   
　　-u 不显示下引号 （根据环境变数 TERM 指定的 terminal 而有所不同）   
　　+/ 在每个档案显示前搜寻该字串（pattern）,然后从该字串之后开始显示   
　　+num 从第 num 行开始显示   
　　fileNames 欲显示内容的档案,可为复数个数   
　　范例：   
　　more -s testfile 逐页显示 testfile 之档案内容,如有连续两行以上空白行则以一行空白行显示。   
　　more +20 testfile 从第 20 行开始显示 testfile 之档案内容。

## mv

使用权限：所有使用者   
　　使用方式：   
　　mv [options] source dest   
　　mv [options] source... directory   
　　说明：将一个档案移至另一档案,或将数个档案移至另一目录。   
　　参数：-i 若目的地已有同名档案,则先询问是否覆盖旧档。   
　　范例：   
　　将档案 aaa 更名为 bbb:  
　　mv aaa bbb   
　　将所有的C语言程式移至 Finished 子目录中:  
　　mv -i \*.c

## rm

使用权限：所有使用者   
　　使用方式：rm [options] name...   
　　说明：删除档案及目录。   
　　把计:   
　　-i 删除前逐一询问确认。   
　　-f 即使原档案属性设为唯读,亦直接删除,无需逐一确认。   
　　-r 将目录及以下之档案亦逐一删除。   
　　范例：   
　　删除所有C语言程式档；删除前逐一询问确认:  
　　rm -i \*.c   
　　将 Finished 子目录及子目录中所有档案删除:  
　　rm -r Finished

## rmdir

名称：rmdir   
　　使用权限：于目前目录有适当权限的所有使用者   
　　使用方式： rmdir [-p] dirName   
　　说明： 删除空的目录。   
　　参数： -p 是当子目录被删除后使它也成为空目录的话,则顺便一并删除。   
　　范例：   
　　将工作目录下,名为 AAA 的子目录删除:  
　　rmdir AAA   
  
　　在工作目录下的 BBB 目录中,删除名为 Test 的子目录。若 Test 删除后,BBB 目录成为空目录,则 BBB 亦予删除。   
　　rmdir -p BBB/Test

## split

使用权限：所有使用者   
　　使用方式：split [OPTION] [INPUT [PREFIX]]说明：   
　　将一个档案分割成数个。而从 INPUT 分割输出成固定大小的档案,其档名依序为 PREFIXaa, PREFIXab...；PREFIX 预设值为 `x。若没有 INPUT 档或为 `-,则从标准输入读进资料。   
　　参数:   
　　-b, --bytes=SIZE   
　　SIZE 值为每一输出档案的大小,单位为 byte。   
　　-C, --line-bytes=SIZE   
　　每一输出档中,单行的最大 byte 数。   
　　-l, --lines=NUMBER   
　　NUMBER 值为每一输出档的列数大小。   
　　-NUMBER   
　　与 -l NUMBER 相同。   
　　--verbose   
　　于每个输出档被开启前,列印出侦错资讯到标准错误输出。   
　　--help   
　　显示辅助资讯然后离开。   
　　--version   
　　列出版本资讯然后离开。   
　　SIZE 可加入单位: b 代表 512, k 代表 1K, m 代表 1 Meg。   
　　范例：  
　　PostgresSQL 大型资料库备份与回存：  
　　因 Postgres 允许表格大过你系统档案的最大容量,所以要将表格 dump 到单一的档案可能会有问题,使用 split进行档案分割。  
　　% pg\_dump dbname | split -b 1m - filename.dump.   
  
　　重新载入  
　　% createdb dbname   
　　% cat filename.dump.\* | pgsql dbname

## touch

名称：touch   
　　使用权限：所有使用者   
　　使用方式：   
　　touch [-acfm]   
　　[-r reference-file] [--file=reference-file]   
　　[-t MMDDhhmm[[CC]YY][.ss]]   
　　[-d time] [--date=time] [--time={atime,access,use,mtime,modify}]   
　　[--no-create] [--help] [--version]   
　　file1 [file2 ...]   
  
　　说明：   
　　touch 指令改变档案的时间记录。 ls -l 可以显示档案的时间记录。  
　　参数：   
　　a 改变档案的读取时间记录。   
　　m 改变档案的修改时间记录。   
　　c 假如目的档案不存在,不会建立新的档案。与 --no-create 的效果一样。   
　　f 不使用,是为了与其他 unix 系统的相容性而保留。   
　　r 使用参考档的时间记录,与 --file 的效果一样。   
　　d 设定时间与日期,可以使用各种不同的格式。   
　　t 设定档案的时间记录,格式与 date 指令相同。   
　　--no-create 不会建立新档案。   
　　--help 列出指令格式。   
　　--version 列出版本讯息。   
　　范例：  
　　最简单的使用方式,将档案的时候记录改为现在的时间。若档案不存在,系统会建立一个新的档案。   
　　touch file   
　　touch file1 file2   
　　将 file 的时间记录改为 5 月 6 日 18 点 3 分,公元两千年。时间的格式可以参考 date 指令,至少需输入 MMDDHHmm ,就是月日时与分。  
　　touch -c -t 05061803 file   
　　touch -c -t 050618032000 file   
　　将 file 的时间记录改变成与 referencefile 一样。  
　　touch -r referencefile file   
  
　　将 file 的时间记录改成 5 月 6 日 18 点 3 分,公元两千年。时间可以使用 am, pm 或是 24 小时的格式,日期可以使用其他格式如 6 May 2000 。   
　　touch -d "6:03pm" file   
　　touch -d "05/06/2000" file   
　　touch -d "6:03pm 05/06/2000" file