# man

## Man page

1. man page的分类



1. man page数据来源

/usr/share/man

可以通过修改配置文件修改man page的搜索路径：/etc/man\_db.conf

1. man用法

* 搜寻特定指令/文件的man page说明文档

例如：跟man这个指令有关的说明文档：man –f/k man

1. whatis 与mandb与man page相关
2. info page:

* info与man的用途其实差不多，都是用来查询指令的用法或者是文件的格式。但是与man page一口气输出一堆信息不同的是，info page则是将文件数据拆成一个一个的段落，每个段落用自己的页面来撰写， 并且在各个页面中还有类似网页的“超链接”来跳到各不同的页面中，每个独立的页面也被称为一个节点（node）。 所以，你可以将info page想成是文字模式的网页显示数据啦！
* 支持info指令的文件默认是放置在/usr/share/info/这个目录当中

1. 其他有用的文件

* /usr/share/doc：指令或者是软件的说明制作成“线上说明文档”
* /usr/share/doc/grub2-tools-2.02：安装须知、预计工作事项、未来工作规划

1. 开关机指令

* 将数据同步写入硬盘中的指令： sync
* 惯用的关机指令： shutdown
* 重新开机，关机： reboot, halt, poweroff

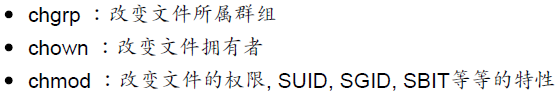
上面谈到的 halt, poweroff, reboot, shutdown 等等，其实都是调用这个systemctl 指令

# Linux文件权限与目录配置

## 文件的权限

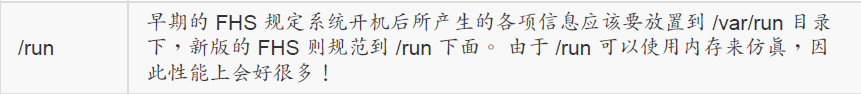
1. 改变文件的权限

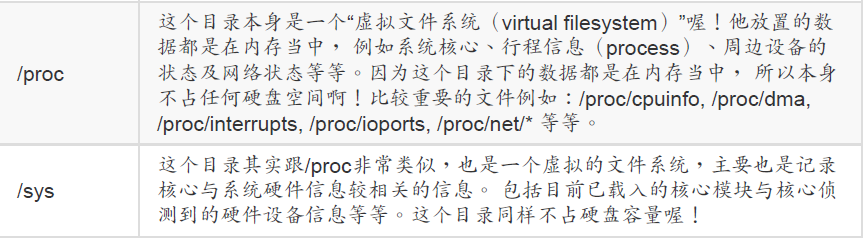
要被改变的群组名称必须要在/etc/group文件内存在才行



## 目录的配置

1. 目录说明

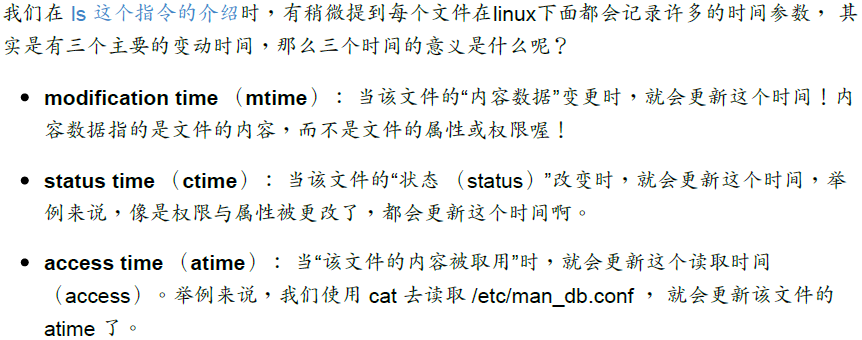




1. 取得路径的文件名称与目录名称

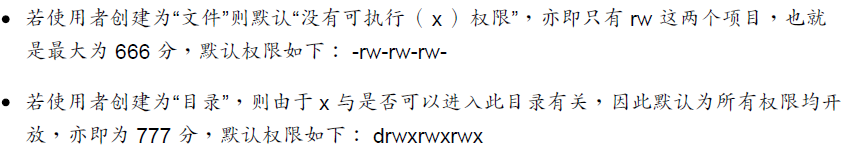
* basename：取得文件名称
* dirname：取得目录名称

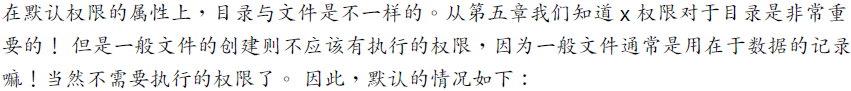
1. 文件的时间



1. 文件的默认权限：umask

* 目前使用者在创建文件或目录时候的权限默认值



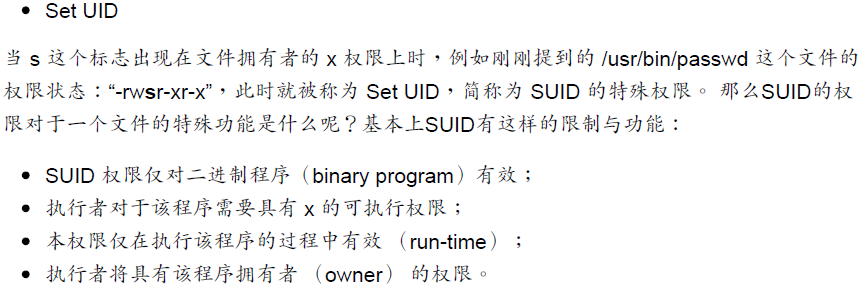


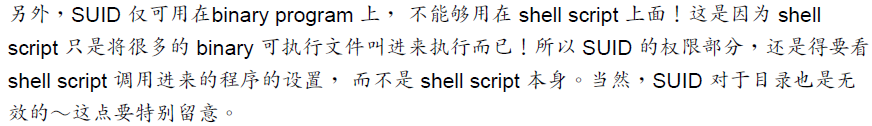


* 关于默认 umask 的设置可以参考 /etc/bashrc 这个文件的内容

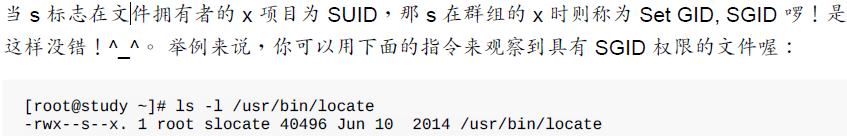
1. 文件特殊权限：SUID, SGID, SBIT

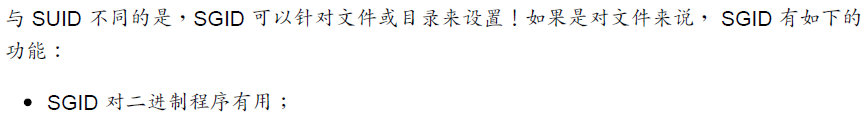
* Set UID

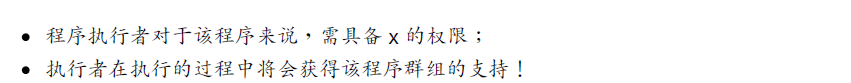


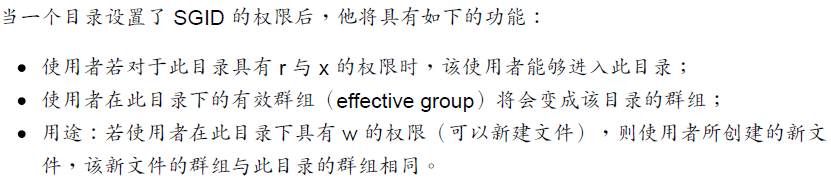


* Set GID

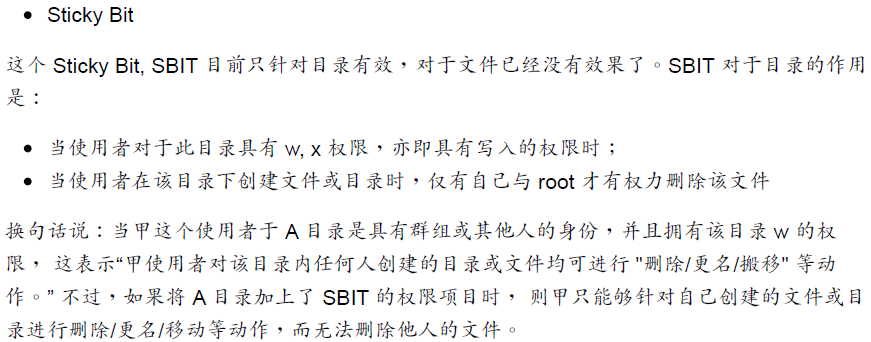




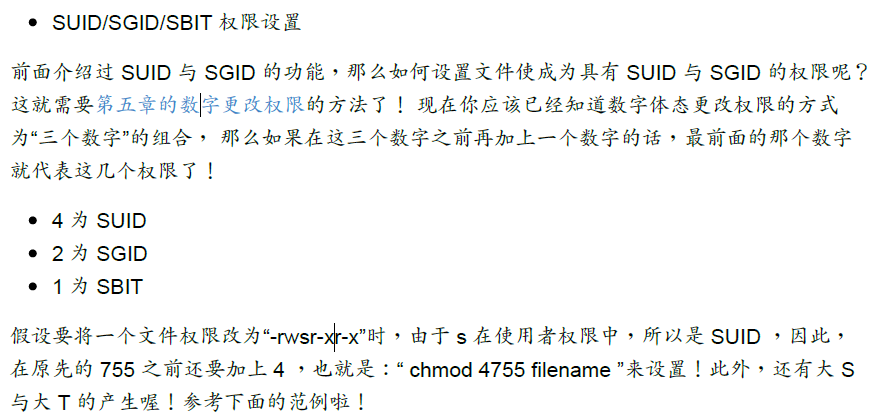


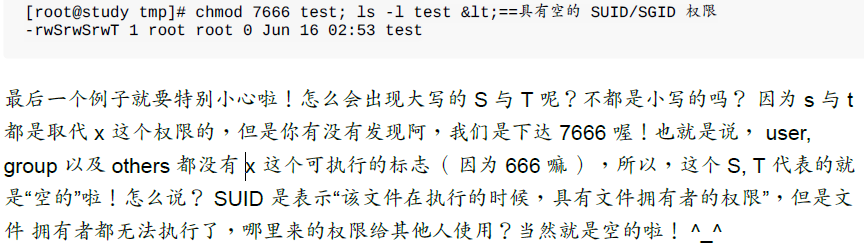


* Sticky Bit



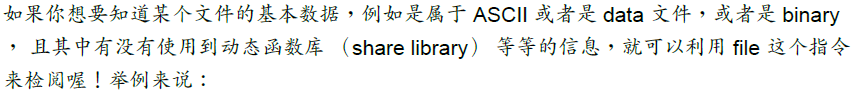
* 权限设置





## 指令与文件的搜索

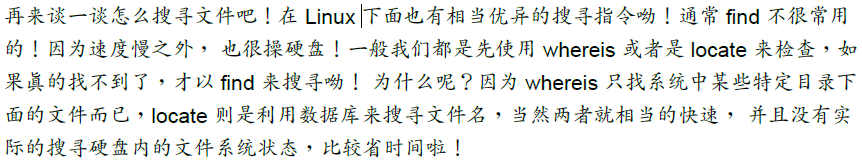
1. 查看文件类型/格式：file



1. 指令文件名的搜寻 which、type



1. 文件文件名的搜寻



* whereis （由一些特定的目录中寻找文件文件名）
* locate / updated

locate 寻找的数据是由“已创建的数据库 /var/lib/mlocate/” 里面的数据所搜寻到的，所以不用直接在去硬盘当中存取数据，

updatedb 指令会去读取 /etc/updatedb.conf 这个配置文件的设置，然后再去硬盘里面进行搜寻文件名的动作， 最后就更新整个数据库文件啰！因为 updatedb 会去搜寻硬盘，所以当你执行 updatedb 时，可能会等待数分钟的时间喔！

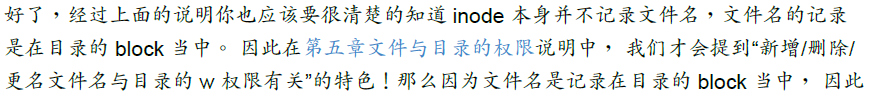
* Find

find [PATH] [option] [action]

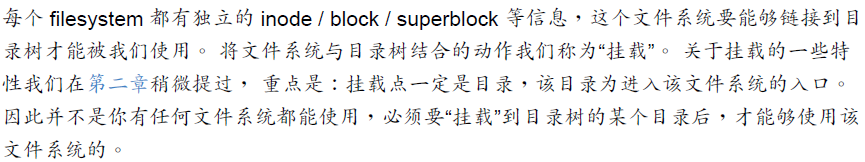
# Linux磁盘与文件系统管理

## 文件系统

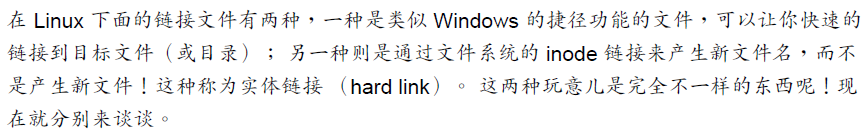
1. dumpe2fs： 查询 Ext 家族 superblock 信息的指令
2. CentOS 7 现在是以 xfs 为默认文件系统

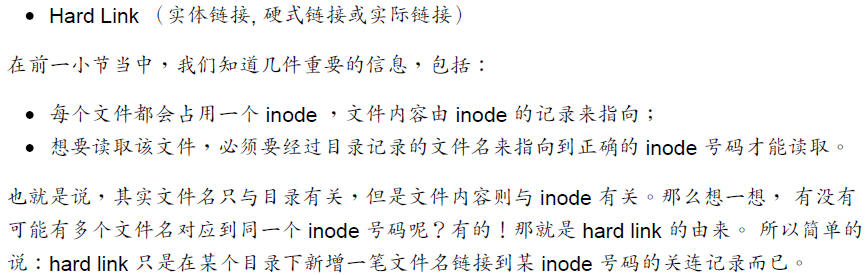


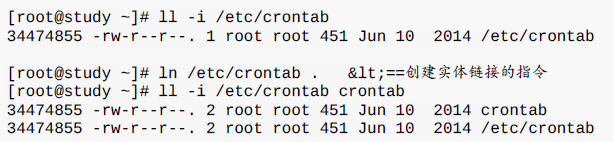
1. 挂载点

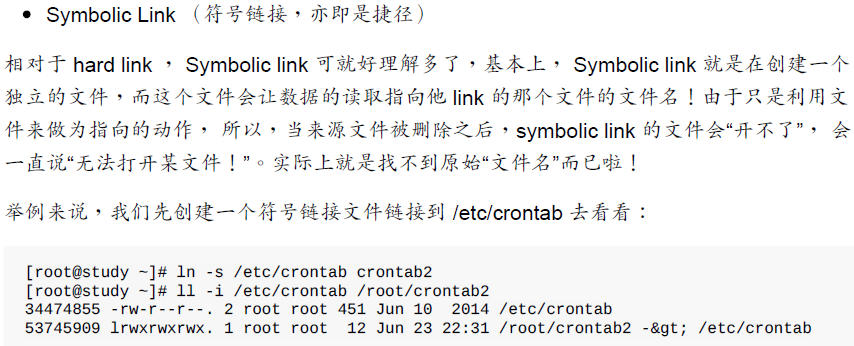


1. 实体链接与符号链接：ln









## 磁盘分区、挂载

1. lsblk 列出系统上的所有磁盘列表

* blkid 列出设备的 UUID 等参数

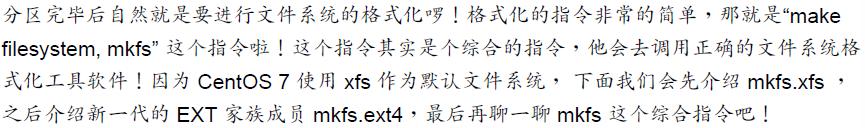
Linux 会将系统内所有的设备都给予一个独一无二的识别码， 这个识别码就可以拿来作为挂载或者是使用这个设备/文件系统之用了。

* parted 列出磁盘的分区表类型与分区信息

1. 磁盘分区： gdisk/fdisk

* MBR 分区表请使用 fdisk 分区， GPT 分区表请使用 gdisk 分区
* partprobe 更新 Linux 核心的分区表信息

## 磁盘格式化（创建文件系统）



1. XFS 文件系统 mkfs.xfs
2. EXT4 文件系统 mkfs.ext4

因为 ext4 的默认值已经相当适合我们系统使用，大部分的默认值写入于我们系统的 /etc/mke2fs.conf 这个文件中

## 文件系统检验

1. EXT4 文件系统 mkfs.ext4

因为 ext4 的默认值已经相当适合我们系统使用，大部分的默认值写入于我们系统的 /etc/mke2fs.conf 这个文件中

1. fsck.ext4 处理 EXT4 文件系统

fsck 是个综合指令

## 文件系统挂载与卸载

1. 挂载点的意义：挂载点是目录， 而这个目录是进入磁盘分区（其实是文件系统啦！）的入口就是了

* 单一文件系统不应该被重复挂载在不同的挂载点（目录）中
* 单一目录不应该重复挂载多个文件系统
* 要作为挂载点的目录，理论上应该都是空目录才是

1. 将文件系统挂载到我们的 Linux 系统上，就要使用 mount 这个指令

* 重新挂载 （remount）
* 可以利用 mount 来将某个目录挂载到另外一个目录去喔！这并不是挂载文件系统，而是额外挂载某个目录的方法！也可以使用 symbolic link 来链接，不过在某些不支持符号链接的程序运行中，还是得要通过这样的方法才行

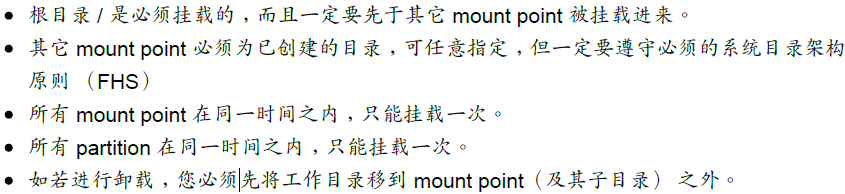
通过这个 mount --bind 的功能， 可以将某个目录挂载到其他目录去喔

1. umount （将设备文件卸载）

## 设置开机挂载

1. 开机挂载 /etc/fstab 及 /etc/mtab在开机的时候就将我要的文件系统都挂好

* /etc/fstab 是开机时的配置文件，不过，实际 filesystem 的挂载是记录到 /etc/mtab 与/proc/mounts 这两个文件当中的。每次我们在更动 filesystem 的挂载时，也会同时更动这两个文件

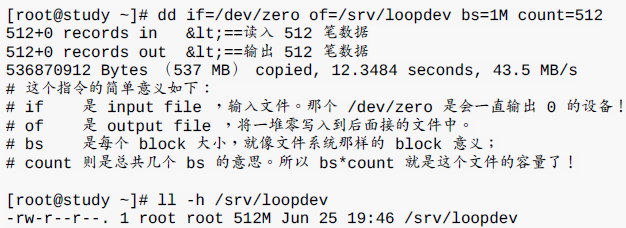


1. 特殊设备 loop 挂载 （镜像文件不烧录就挂载使用）

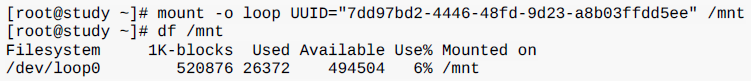
* 如果有光盘镜像文件，或者是使用文件作为磁盘的方式时，那就得要使用特别的方法来将他挂载起来，不需要烧录



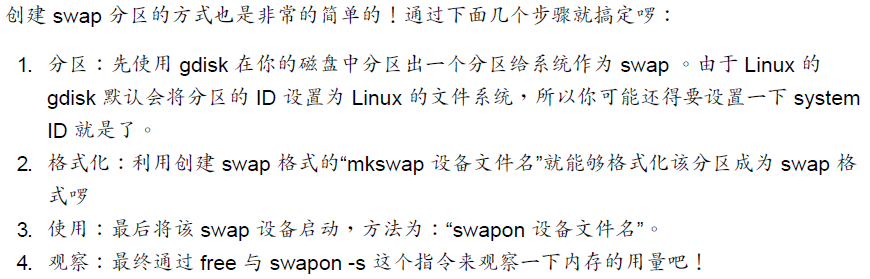
* 要作为挂载点的目录，理论上应该都是空目录才是
* 创建大文件以制作 loop 设备文件







1. 使用实体分区创建swap



1. 使用文件创建swap
2. parted 可以直接在一行命令行就完成分区，是一个非常好用的指令
3. mkisofs:创建镜像文件

* 新版的 CentOS 7 使用的是 wodim 这个文字界面指令来进行烧录的行为
* dd 可以读取磁盘设备的内容（几乎是直接读取扇区"sector"），然后将整个设备备份成一个文件
* cpio 可以备份任何东西，包括设备设备文件。不过 cpio 有个大问题， 那就是 cpio 不会主动的去找文件来备份！一般来说， cpio 得要配合类似 find 等可以找到文件名的指令来告知 cpio 该被备份的数据在哪里

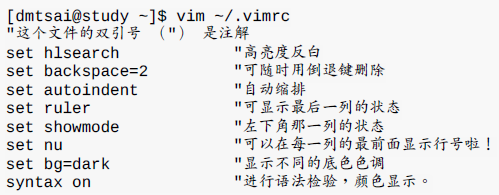
# vim

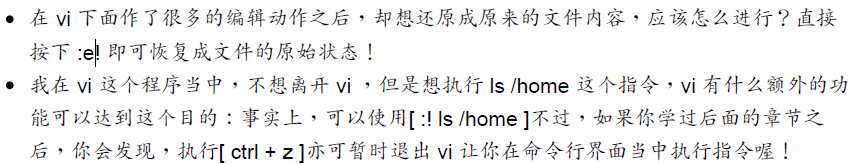
## 挑字补全功能



## vim环境设置与记录：~/.vimrc, ~/.viminfo

1. 整体 vim 的设置值一般是放置在 /etc/vimrc 这个文件，不过，不建议你修改他！ 你可以修改 ~/.vimrc 这个文件 （默认不存在，请你自行手动创建！），将你所希望的设置值写入



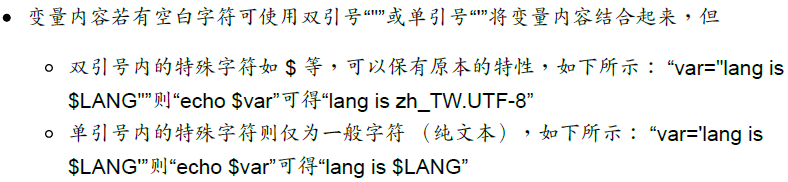


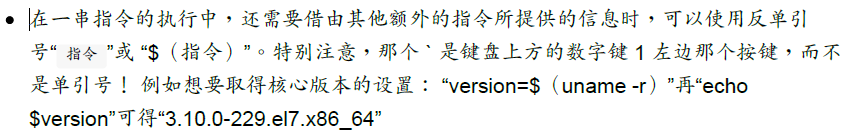
1. Linux 系统默认支持的语系数据：这与 /etc/locale.conf 有关

# bash

## shell

1. /etc/shells 这个文件，至少就有下面这几个可以用的 shells



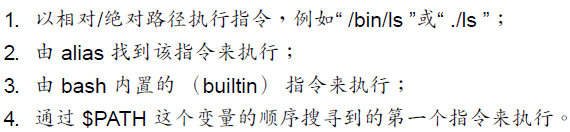


1. 查看Linux的核心版本

uname -r

## 与文件系统及程序的限制关系：ulimit

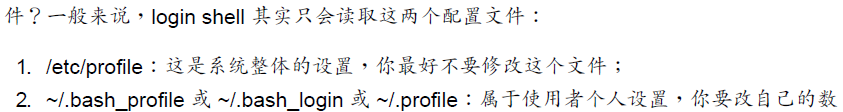
1. ulimit：bash 是可以“限制使用者的某些系统资源”的，包括可以打开的文件数量， 可以使用的 CPU 时间，可以使用的内存总量等等
2. 路径与指令搜索顺序



先 alias 再 builtin 再由 $PATH 找

## Login shell

1. Login shell 读取的配置文件

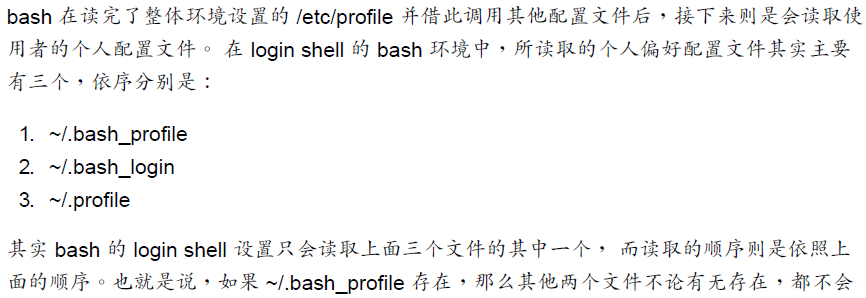


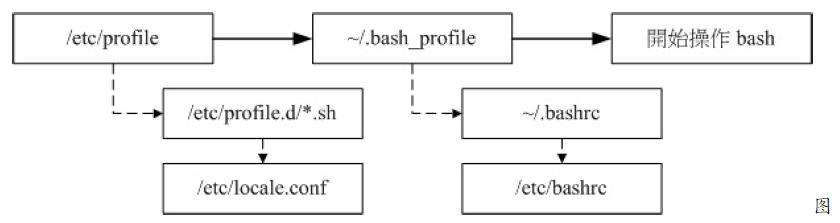


1. /etc/profile调用的脚本

* /etc/profile.d/\*.sh
* /etc/locale.conf
* /usr/share/bash-completion/completions/\*

1. ~/.bash\_profile （login shell 才会读）





* ~/.bash\_profile 其实会再调用 ~/.bashrc 的设置内容，在 CentOS 的 login shell 环境下，最终被读取的配置文件是“ ~/.bashrc ”这个文件喔！所以，可以将自己的偏好设置写入该文件即可
* 直接读取配置文件而不登出登陆呢？ 可以的！那就得要利用 source 这个指令

## non-login shell

1. ~/.bashrc （non-login shell 会读）

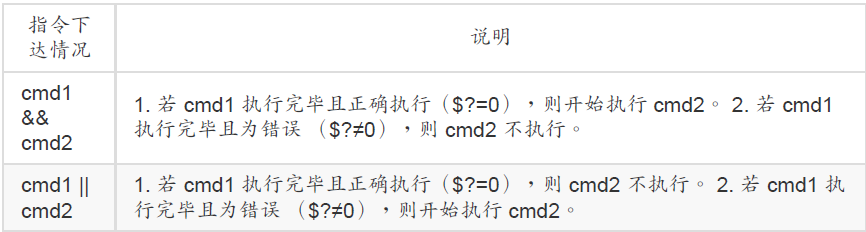
## 数据流重导向

1. 数据流重导向可以将 standard output （简称 stdout） 与 standard error output （简称 stderr） 分别传送到其他的文件或设备去，而分别传送所用的特殊字符则如下所示

* 标准输入　　（stdin） ：代码为 0 ，使用 <（输入） 或 <<（结束输入）
* 标准输出　　（stdout）：代码为 1 ，使用 > 或 >>
* 标准错误输出（stderr）：代码为 2 ，使用 2> 或 2>>
* 写入同一个文件的特殊语法如上表所示，你可以使用 2>&1
* 既然有 2>&1 来将 2> 转到 1> 去，那么应该也会有 1>&2

1. /dev/null 垃圾桶黑洞设备与特殊写法
2. 命令执行的判断依据

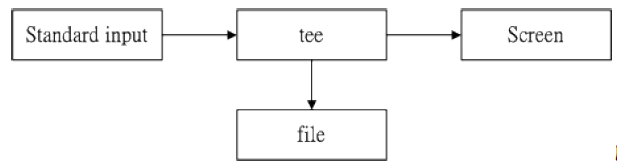
* cmd ; cmd （不考虑指令相关性的连续指令下达）



1. 管线命令“ | ”仅能处理经由前面一个指令传来的正确信息

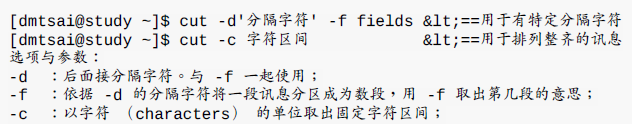
* 在每个管线后面接的第一个数据必定是“指令”喔！而且这个指令必须要能够接受 standardinput 的数据才行，这样的指令才可以是为“管线命令”

1. 双向重导向： tee

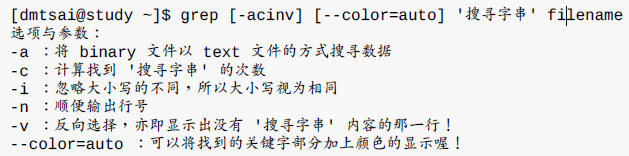


## 撷取命令： cut, grep

1. cut 主要的用途在于将“同一行里面的数据进行分解

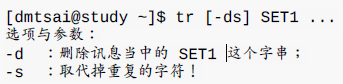


1. grep 则是分析一行讯息， 若当中有我们所需要的信息，就将该行拿出来



## 撷取命令： cut, grep

1. 排序命令： sort, wc, uniq
2. 字符转换命令： tr, col, join, paste, expand



* join 看字面上的意义 （加入/参加） 就可以知道，他是在处理两个文件之间的数据， 而且，主要是在处理“两个文件当中，有 "相同数据" 的那一行，才将他加在一起”的意

## 字符转换命令： tr, col, join, paste, expand

1. tr 可以用来删除一段讯息当中的文字，或者是进行文字讯息的替换
2. col可以用来简单的处理将 [tab] 按键取代成为空白键
3. join在处理两个文件之间的数据， 而且，主要是在处理“两个文件当中，有 "相同数据" 的那一行，才将他加在一起”的意思
4. paste 就直接“将两行贴在一起，且中间以 [tab] 键隔开”而已
5. expand将 [tab] 按键转成空白键, unexpand 这个将空白转成 [tab] 的指令
6. 分区命令： split. 可以帮你将一个大文件，依据文件大小或行数来分区，就可以将大文件分区成为小文件
7. 参数代换： xargs. xargs 可以读入 stdin 的数据，并且以空白字符或断行字符作为分辨，将 stdin 的数据分隔成为 arguments
8. 正则表达式就是处理字串的方法，他是以行为单位来进行字串的处理行为， 正则表达式通过一些特殊符号的辅助，可以让使用者轻易的达到“搜寻/删除/取代”某特定字串的处理程序