# 工作管理

### 将工作丢到背景中

- 使用&，将某条指令放到背景去运行
  **数据流重导向(???)**
- ctrl+z
  使用ctrl+z丢到背景当中的工作都是暂停状态。

### 观察目前的背景工作状态

- jobs
  -l: 除了列出job number 与指令串之外，同时列出PID的号码
  -r: 仅列出正在背景run的工作
  -s: 仅列出正在背景当中暂停(stop)的工作

### 提示

'+'表示最近一个被丢进背景的工作，且目前在背景下预设会被取用的那个工作(与fg指令相关)。
'-'表示最后第二个被丢进背景的工作

### 将背景工作拿到前景来处理

- fg %jobnumber
  %jobnumber: jobnumber为工作号码。%可有可无

### 让工作在背景下的状态变成运作中

- bg
  用法和fg差不多，使用%jobnumber.

### 管理背景当中的工作

- kill
  kill -signal %jobnumber
  kill -l(L的小写)：列出目前kill能够使用的讯号signal有哪些。
  -1: 重新读取一次参数的配置工作
  -2：代表与由键盘驶入ctrl+c同样的工作
  -9：立即强制删除一个工作
  -15：以正常的进程方式终止一项工作，为默认值。

# 进程管理

### 进程的观察

- ps: 将某个时间点的进程运作情况截取下来
  -A: 所有的process均显示出来
  -a: 不与terminal有关的所有process
  -u: 有效使用者(effective user)相关的process
  x: 通常与a这个参数一起使用，可列出较完整信息
  输出格式规划：
  l: 较长、较详细的将该PID的信息列出
  j: 工作的格式
  -f: 做一个更完整的输出。
  记住两个：
  ps -l: 只能查阅自己bash进程
  ps aux: 可以查阅所有系统运作的进程

ps展示的进程信息：

> F: 代表这个进程旗标，说明这个进程的总结权限，常用的号码：
>
> > 4 表示此进程的权限为root
> > 1 表示此子进程仅进行复制(fork)而没有实际执行。

> S: 代表这个进程的状态
>
> > R(Running): 该程序正在运作中
> > S(Sleep): 该程序目前正在睡眠状态，但可以被唤醒
> > D: 不可被唤醒的睡眠状态，可能在等待I/O
> > T: 停止状态，可能是在工作控制(背景暂停)或除错状态
> > Z: 僵尸状态，进程已经终止但却无法被移除至内存外

- top: 动态观察进程的变化
  -d: 后面可以接秒数，就是整个进程画面更新的秒数，预设是5秒
  -b: 以批次的方式进行top
  -n: 与-b搭配，需要进行几次top的输出结果
  -p: 指定某些个PID来进行观察检测而已
- pstree: 查询进程的相关性

**进程的管理**

- kill -signal PID
- killall -signal [command name] (不大会)
  -i: 交互式的，若需要删除时，会出现提示字符给用户
  -e: exact的意思，表示后面接的command name要一致，但这整个完整的指令不能超过15个字符。
  -I: 指令名称忽略大小写。

**关于进程的执行顺序**

linux给予进程一个所谓的[优先执行序(priority, PRI)]，PRI值越低代表越优先，不过这个PRI值是由核心动态调整，用户无法直接调整PRI值。
想要调整进程的PRI时，需要通过Nice值，也就是NI。

- nice：新执行的指令即给与新的nice值
  -n: 后面接一个数值，数值的范围为-20~19
- renice: 已存在进程的nice重新调整
  renice [number] PID

**系统资源的观察**

- free: 观察内存使用情况
  -b/m/k/g: 使用bytes/Mbytes/Kbytes/Gbytes显示
  -t: 在输出的最终结果，显示物理内存与swap的总量
  -s: 可以让系统每几秒钟输出一次
  -c: 与-s同时处理
- uname: 查阅系统与核心相关信息
  -a: 所有系统相关的信息，包括底下的数据都会被列出来
  -s: 系统核心名称
  -r: 核心的版本
  -m: 本系统的硬件名称
  -p: CPU的类型
  -i: 硬件平台