1. 目录

我们可以把目录当做文件来看，inode中存储是该目录的属性，block中存储的该目录中包含的文件的inode的位置。
使用ls -li 命令可以查出当前文件夹下目录的所占用的inode号码。

block的基本大小是4k，则其他任何文件或目录的大小都是4k的倍数。

2. 文件

每当建立一个文件是，系统会分配1个inode，和与之大小想匹配的block数量；
假设1个block的小时4k，当存储1个100k的文件时，需要25个block，然而一个inode仅有12个直接指针，因此额外需要1个block作为间接的存储。

经过上面的说明你也应该要很清楚的知道 inode 本身并不记录文件名，文件名的记录是在目
录的 block 当中。因此才会提到『新增/删除/更名文件名与目录的 w 权限有关』

3. 目录树读取

目录的读取，其实是从根目录/ 一层层的读取的，先找inode，再找block，再找下一层目录的。。。

当文件系统中文件进行来来回回的读写，会造成碎片文件，即便文件的block是填入式的装填，也可能会造成硬盘的机械手臂频繁的移动，最好的解决办法就是将数据备份，格式化硬盘后，重新拷贝进去。

4. 异步处理

如果你常常编辑一个好大的文件， 在编辑的过程中又频繁的要系统来写入到磁盘中，由于磁盘写入的速度要比内存慢很多， 因此你会常常耗在等待磁盘的写入/读取上。真没效率！
为了解决这个效率的问题，因此我们的 Linux 使用的方式是透过一个称为异步处理(asynchronously)的方式。所谓的异步处理是这样的：
当系统加载一个文件到内存后，如果该文件没有被更动过，则在内存区段的文件数据会被设定为干净
(clean)的。 但如果内存中的文件数据被更改过了(例如你用 nano 去编辑过这个文件)，此时该内存中的数据会被设定为脏的 (Dirty)。此时所有的动作都还在内存中执行，并没有写

入到磁盘中！ 系统会不定时的将内存中设定为『Dirty』的数据写回磁盘，以保持磁盘与内存数据的一致性。 你也可以利用第四章谈到的 sync 指令来手动强迫写入磁盘。

系统会将常用的文件数据放置到主存储器的缓冲区，以加速文件系统的读/写；

- **承上，因此 Linux 的物理内存最后都会被用光！这是正常的情况！可加速系统效能；**
- 你可以手动使用 sync 来强迫内存中设定为 Dirty 的文件回写到磁盘中；
- 若正常关机时，关机指令会主动呼叫 sync 来将内存的数据回写入磁盘内；
- 但若不正常关机(如跳电、当机或其他不明原因)，由于数据尚未回写到磁盘内， 因此重新启动后可能会花

很多时间在进行磁盘检验，甚至可能导致文件系统的损毁(非磁盘损毁)。

5. 文件挂载

所谓的文件挂载其实是将某个文件系统与目录树结合起来。挂载点一定是目录，进入该文件系统的入口就是该目录。