# LINUX应用编程和网络编程（一）

## 什么是文件IO

IO就是input/output,输入/输出。文件IO的意思就是读写文件

## 文件操作的主要接口API

### 什么是【操作系统API】

API[应用程序接口]是一些【函数】

### Linux常用文件IO接口

open close write read lseek(移动文件指针)

### 文件操作的一般步骤

1）在Linux系统中要操作一个文件，一般是先open开一个文件，得到一个【文件描述符】，然后对文件进行读写操作（或其它操作），最后close关闭文件即可

2）强调一点：我们对文件进行操作时，一定要先打开文件，打开成功后才能去操作（如果打开本身失败，后面就不用操作了）；最后读写完成之后一定要close关闭文件，否则可能会造成文件损坏

3）文件平时是存在【块设备】中的文件系统中的，我们把这种文件 叫【静态文件】。当我们去open打开一个文件；内核在内存中申请一段内存，并且将静态文件的内容从块设备中读取到内存中【特定地址，又操作系统分配】管理存放（叫动态文件）。

**小结：**静态文件就是平时存储在硬盘中没有被【打开】时候的文件，动态文件就是从块设备中读取到内存中的文件

4）【打开文件】后，以后对这个文件的读写操作，都是针对内存中这一份动态文件的，而并不是针对静态文件的。当我们对动态文件进行读写后，此时内存中的动态文件和块设备中的静态文件就不同步了，当我们close关闭动态文件时，close内部内核将内存中的动态文件的内容去更新（同步）块设备中的静态文件。（这也就是为什么我们要定期保存的原因【为了实时更新同步】）

5）常见的一些现象：

第一个：打开一个大文件时比较慢（因为内核要把【整个文件】从块设备读取到内存中，并且建立数据结构对其进行管理）

第二个：我们写了一半的文件，如果没有点保存直接关机/断电，重启后文件内容丢失。

6）为什么要这么设计（在内存中为什么要重新搞一份，而不是直接对块设备中的文件进行读写操作）？

因为块设备本身有读写限制（回忆NnadFlash,SD等块设备的读写特征，只能读不能写），本身对块设备进行操作非常不灵活（因为只能按照块来读写）。而内存可以按【字节为单位】来操作，而且可以随机操作（内存就叫RAM，random,可以随意随机指定一个地址去操作），很灵活。所以内核设计文件操作时就这么设计了，增强了文件操作的灵活性

### 重要概念：文件描述符