文件描述符合打开文件之间的关系

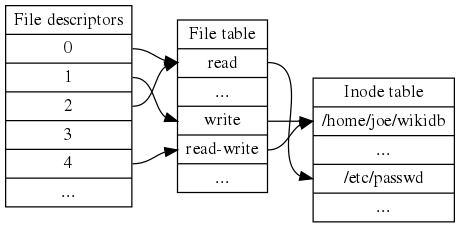
每一个文件描述符会与一个打开文件相对应，同时，不同的文件描述符也会指向同一个文件。相同的文件可以被不同的进程打开也可以在同一个进程中被多次打开。系统为每一个进程维护了一个文件描述符表，该表的值都是从0开始的，所以在不同的进程中你会看到相同的文件描述符，这种情况下相同文件描述符有可能指向同一个文件，也有可能指向不同的文件

三个数据结构（都由内核维护）：

File descriptors：每个进程对应一个

File table: 全系统一个, 每条记录代表打开文件（或其他资源）的模式：读取，写入，追加以及可能的其他模式

Inode table: 全系统一个



In the traditional implementation of Unix, file descriptors index into a per-process file descriptor table maintained by the kernel, that in turn indexes into a system-wide table of files opened by all processes, called the file table. This table records the mode with which the file (or other resource) has been opened: for reading, writing, appending, and possibly other modes. It also indexes into a third table called the inode table that describes the actual underlying files.[3] To perform input or output, the process passes the file descriptor to the kernel through a system call, and the kernel will access the file on behalf of the process. The process does not have direct access to the file or inode tables.

On Linux, the set of file descriptors open in a process can be accessed under the path /proc/PID/fd/, where PID is the process identifier.