<http://blog.sina.com.cn/s/blog_533074eb0101173o.html>

fread返回的是一个FILE结构指针  
而read返回的是一个int的文件号  
  
前者fopen/fread的实现是靠调用底层的open/read来实现的.  
  
fopen/fread  
是C标准的库函数，操作的对象是： file stream  
  
open/read  
是和操作系统有关的系统调用。操作的对象是： “file descriptor”  
  
  
f是ANSI的C标准库。后面的是UNIX下的系统调用。  
  
带f的带有缓冲，是后面的衍生，  
直接和硬件打交道，必须是后面的！

UNIX环境下的C 对二进制流文件的读写有两套班子：1) fopen,fread,fwrite ; 2) open, read, write  
这里简单的介绍一下他们的区别。  
1. fopen 系列是标准的C库函数；open系列是 POSIX 定义的，是UNIX系统里的system call。  
也就是说，fopen系列更具有可移植性；而open系列只能用在 POSIX 的操作系统上。  
2. 使用fopen 系列函数时要定义一个指代文件的对象，被称为“文件句柄”（file handler），是一个结构体；而open系列使用的是一个被称为“文件描述符” （file descriptor）的int型整数。  
3. fopen 系列是级别较高的I/O，读写时使用缓冲；而open系列相对低层，更接近操作系统，读写时没有缓冲。由于能更多地与操作系统打交道，open系列可以访问更改一些fopen系列无法访问的信息，如查看文件的读写权限。这些额外的功能通常因系统而异。  
4. 使用fopen系列函数需要"#include <sdtio.h>"；使用open系列函数需要"#include <fcntl.h>" ，链接时要之用libc（-lc）  
小结：  
总的来说，为了使程序获得更好的可移植性，未到非得使用一些fopen系列无法实现的功能的情况下，fopen系列是首选。

**read/write和fread/fwrite区别**

1,fread是带缓冲的,read不带缓冲.

eg:

如果文件的大小是8k。你如果用read/write，且只分配了2k的缓存，则要将此文件读出需要做4次系统调用来实际从磁盘上读出。  
如果你用fread/fwrite，则系统自动分配缓存，则读出此文件只要一次系统调用从磁盘上读出。  
也就是用read/write要读4次磁盘，而用fread/fwrite则只要读1次磁盘。效率比read/write要高4倍。

如果程序对内存有限制，则用read/write比较好  
2,fopen是标准c里定义的,open是POSIX中定义的.  
3,fread可以读一个结构. read在linux/unix中读二进制与普通文件没有区别.  
4,fopen不能指定要创建文件的权限.open可以指定权限.  
5,fopen返回指针,open返回文件描述符(整数).  
6,linux/unix中任何设备都是文件,都可以用open,read.  
  
都用fread 和fwrite,它自动分配缓存,速度会很快,比自己来做要简单。**如果要处理一些特殊的描述符,用read 和write,如套接口,管道之类的**系统调用write的效率取决于你buf的大小和你要写入的总数量，**如果buf太小，你进入内核空间的次数大增，效率就低下**。**而fwrite会替你做缓存，减少了实际出现的系统调用，所以效率比较高。**  
如果只调用一次(可能吗?)，这俩差不多，严格来说write要快一点点(因为**实际上fwrite最后还是用了write做真正的写入文件系统工作**)，但是这其中的差别无所谓。