文件I/O（一）

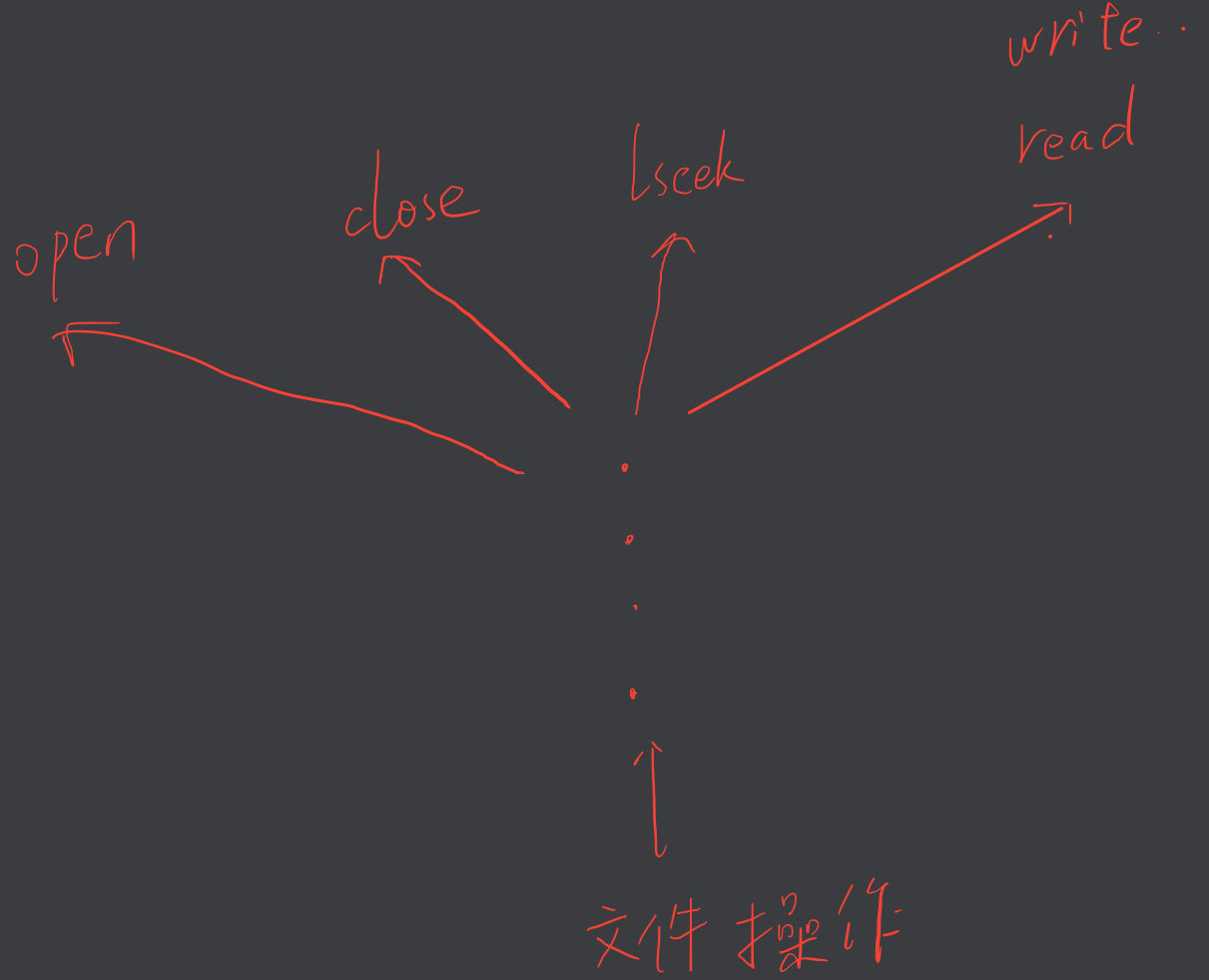
主讲：大海老师

课程目标

如何理解文件IO（了解）

文件描述符的含义（了解）

小结



文件I/O – 介绍

什么是文件I/O?

posix(可移植操作系统接口)定义的一组函数

不提供缓冲机制，每次读写操作都引起系统调用

核心概念是文件描述符

访问各种类型文件

Linux下, 标准IO基于文件IO实现

以 FILE 就是个对象 名. 起止 操作

描述符 fd 的就是其数组下标吧

	标准ＩＯ	文件ＩＯ(低级IO)
打开	fopen,freopen,fdopen	open
关闭	fclose	close
读	getc,fgetc,getchar fgets,gets fread	read
写	putc,fputc,putchar fputs,puts, fwrite	write

文件I/O – 文件描述符

- 每个打开的文件都对应一个文件描述符。
- 文件描述符是一个非负整数。Linux为程序中每个打开的文件分配一个文件描述符。
- 文件描述符从0开始分配，依次递增。
- 文件IO操作通过文件描述符来完成。
- 0, 1, 2 的含义?

文件I/O – open

open函数用来创建或打开一个文件:

```
#include <fcntl.h>
int open(const char *pathname, int flags);
int open(const char *pathname, int flags, mode_t
    mode);
```

成功时返回文件描述符；出错时返回EOF

- 打开文件时使用两个参数
- 创建文件时第三个参数指定新文件的权限，（只有在建立新文件时有效）此外真正建文件时的权限会受到umask 值影响，实际权限是mode-umaks
- 可以打开设备文件，但是不能创建设备文件（创建设备mknode 驱动部分会讲）

文件I/O – open

原型	int open(const char *pathname, int flags, mode_t mode);		
参数	pathname	被打开的文件名（可包括路径名）。	
	flags	O_RDONLY：只读方式打开文件。	这三个参数互斥
		O_WRONLY：可写方式打开文件。	
		O_RDWR：读写方式打开文件。	
		O_CREAT：如果该文件不存在，就创建一个新的文件，并用第三的参数为其设置权限。	
		O_EXCL：如果使用O_CREAT时文件存在，则可返回错误消息。这一参数可测试文件是否存在。	
		O_NOCTTY：使用本参数时，如文件为终端，那么终端不可以作为调用open()系统调用的那个进程的控制终端。	
		O_TRUNC：如文件已经存在，那么打开文件时先删除文件中原有数据。	
		O_APPEND：以添加方式打开文件，所以对文件的写操作都在文件的末尾进行。	
	mode	被打开文件的存取权限，为8进制表示法。	

文件I/O – open

r	O_RDONLY
r+	O_RDWR
w	O_WRONLY | O_CREAT | O_TRUNC, 0664
w+	O_RDWR | O_CREAT | O_TRUNC, 0664
a	O_WRONLY | O_CREAT | O_APPEND, 0664
a+	O_RDWR | O_CREAT | O_APPEND, 0664

文件I/O – open

- umask ：用来设定文件或目录的初始权限
- 文件和目录的真正初始权限
- 文件或目录的初始权限 = 文件或目录的最大默认权限 - umask权限

文件I/O – open – 示例1

以只写方式打开文件1.txt。如果文件不存在则创建，如果文件存在则清空：

```
int  fd;

if ((fd  = open( “1.txt” , O_WRONLY|O_CREAT|O_TRUNC, 0666)) < 0) {
    perror( “open” );
    return  -1;
}
……
```

文件I/O – open – 示例2

以读写方式打开文件1.txt。如果文件不存在则创建，

如果文件存在则报错：

```
int  fd;

if ((fd  = open( “1.txt” , O_RDWR|O_CREAT|O_EXCL, 0666)) < 0) {
    if (errno == EEXIST) {
        perror( “exist error” );
    } else {
        perror( “other error” );
    }
}
```

文件I/O – close

close函数用来关闭一个打开的文件:

```
#include  <unistd.h>
int  close(int fd);
```

- 成功时返回0；出错时返回EOF
- 程序结束时自动关闭所有打开的文件
- 文件关闭后，文件描述符不再代表文件

文件I/O – 小结

文件描述符

open

close