Day1 学习要点

**概述：**

* Linux IO进程课程目的：学习编写linux应用程序（APP）
* Linux文件的种类 :

常规文件

目录文件

字符文件

块文件

链接文件（相当于windows快捷方式）

* IO的概念

I input 输入设备 比如键盘鼠标都是Input设备

O output 输出设备 比如显示器

优盘，网口，既是输入也是输出

* 系统调用和库函数

系统调用就是操作系统提供的接口函数.

如果我们把系统调用封装成库函数就可以起到隔离的作用，提供程序的可移植性。

Printf就是库函数然后调用了系统调用才在显示器上显示字符。

* 流的概念

就是数据的流，在程序中就是一个结构体。

* Windows 和linux的换行符区别

Windows是\r\n

Linux 是\n

* 缓冲区的概念

为了减少操作IO设备的次数，提高运行效率，在内存里面设置的缓冲区，

全缓冲：缓冲区满才输出

行缓冲：遇到换行符输出

* 三种标准IO ：

|  |  |  |  |
| --- | --- | --- | --- |
| 标准输入流（键盘） | 0 | STDIN\_FILENO | stdin |
| 标准输出流（显示器） | 1 | STDOUT\_FILENO | stdout |
| 标准错误流 | 2 | STDERR\_FILENO | stderr |

* 缓冲区概念演示：

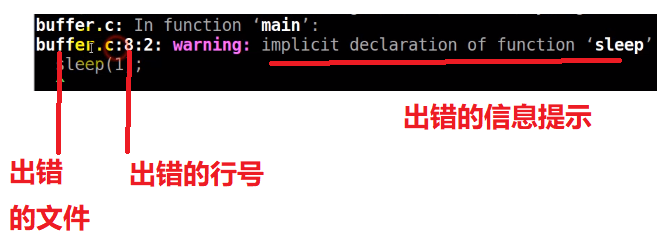
Buffer.c 代码

程序正常结束会刷新缓冲区。

* Sleep函数：是释放cpu给其他应用程序使用的库函数。使用的头文件是#include <unistd.h>

查看头文件方法：man 2 函数 ，或者 man 3 函数

* 编译错误告警查看方法



**文件的打开和关闭概念**

打开就是占用资源

关闭就是释放资源

**文件的打开**

* 文件的打开函数

FILE \*fopen (const char \*path, const char \*mode);

Path: 普通文件当前路径不需要加目录，其他要使用完整的路径

Mode：

返回值：出错返回NULL，所以使用fopen函数必须判断是否为空

* 文件打开的模式（非常重要）

|  |  |
| --- | --- |
| “r” 或 “rb” | 以只读方式打开文件，文件必须存在。 |
| “r+” 或 ”r+b” | 以读写方式打开文件，文件必须存在。 |
| “w” 或 “wb” | 以只写方式打开文件，若文件存在则文件长度清为0。若文件不存在则创建。 |
| “w+” 或 “w+b” | 以读写方式打开文件，其他同”w”。 |
| “a” 或 “ab” | 以只写方式打开文件，若文件不存在则创建；向文件写入的数  据被追加到文件末尾。 |
| “a+” 或 “a+b” | 以读写方式打开文件。其他同”a” |

* 编译错误：

f\_open.c:9:38: error: ‘errno’ undeclared (first use in this function)

printf("fopen:%s\n",strerror(errno));

error: ‘errno’ undeclared 表示errno变量没有定义

解决方法：如果是系统变量用include 头文件，如果是你自己的，自己手动定义。

f\_open.c:10:29: warning: implicit declaration of function ‘strerror’ [-Wimplicit-function-declaration]

printf("fopen:%s\n",strerror(errno));

warning: implicit declaration of function ‘strerror’ 表示strerror函数隐示的声明

解决方法：include 添加对应的头文件。

* perror 库函数 头文件stdio.h

strerror 库函数 头文件 errno.h string.h

perror和strerror 功能：打印系统的错误描述（注意：是系统错误，不是你自己代码错误）

**文件的关闭：**

函数原型：int fclose(FILE \*stream)

* fclose()调用成功返回0，失败返回EOF（-1），并设置errno
* 流关闭时自动刷新缓冲中的数据并释放缓冲区，比如：常规文件把缓冲区内容写入磁盘
* 当一个程序正常终止时，所有打开的流都会被关闭
* fclose()函数的入参stream必须保证为非空，否则出现断错误。