

# 实验四 Linux文件目录

## 一、实验目的

- 1、了解Linux文件系统与目录操作；
- 2、了解Linux文件系统目录结构；
- 3、掌握文件和目录的程序设计方法。

## 二、实验内容

编程实现目录查询功能：

- 功能类似ls -lR;
- 查询指定目录下的文件及子目录信息;
- 显示文件的类型、大小、时间等信息;
- 递归显示子目录中的所有文件信息。

## 三、预备知识

### 1、Linux文件属性接口

```
#include <unistd.h>
```

```
#include <sys/stat.h>
```

```
#include <sys/types.h>
```

```
int fstat(int fildes, struct stat *buf);
```

返回文件描述符相关的文件的状态信息

```
int stat(const char *path, struct stat *buf);
```

通过文件名获取文件信息，并保存在buf所指的结构体stat中

```
int lstat(const char *path, struct stat *buf);
```

如读取到了符号连接，lstat读取符号连接本身的状态信息，而stat读取的是符号连接指向文件的信息。



```
struct stat {  
    unsigned long  st_dev;      // 文件所属的设备  
    unsigned long  st_ino;      // 文件相关的inode  
    unsigned short st_mode;     // 文件的权限信息和类型信息:  
                                S_IFDIR, S_IFBLK, S_IFIFO, S_IFLINK  
    unsigned short st_nlink;    // 硬连接的数目  
    unsigned short st_uid;      // 文件所有者的ID  
    unsigned short st_gid;      // 文件所有者的组ID  
    unsigned long  st_rdev;     // 设备类型  
    unsigned long  st_size;     // 文件大小  
    unsigned long  st_blksize;  // 块大小  
    unsigned long  st_blocks;   // 块数  
    unsigned long  st_atime;    // 文件最后访问时间  
    unsigned long  st_atime_nsec;  
    unsigned long  st_mtime;    // 最后修改内容的时间  
    unsigned long  st_mtime_nsec;  
    unsigned long  st_ctime;    // 文件最后修改属性的时间  
    unsigned long  st_ctime_nsec;  
    unsigned long  __unused4;  
    unsigned long  __unused5;  
};
```

stat结构体几乎保存了所有的文件状态信息

## 2、Linux目录结构接口

```
#include <sys/types.h>
```

```
#include <dirent.h>
```

```
#include <unistd.h>
```

- **opendir()**

DIR \*opendir(const char \*name);

通过路径打开一个目录，返回一个DIR结构体指针（目录流），失败返回NULL；

- **readdir()**

struct dirent \*readdir(DIR \*)

读取目录中的下一个目录项，没有目录项可以读取时，返回为NULL；

## 目录项结构:

```
struct dirent {  
    #ifndef __USE_FILE_OFFSET64  
        __ino_t d_ino; //索引节点号  
        __off_t d_off; //在目录文件中的偏移  
    #else  
        __ino64_t d_ino;  
        __off64_t d_off;  
    #endif  
    unsigned short int d_reclent; //文件名的长度  
    unsigned char d_type; //d_name所指的文件类型  
    char d_name[256]; //文件名  
};
```

注：需跳过两个目录项 “.”和 “..”  
定义见/usr/include/dirent.h

- `chdir()`

`int chdir(const char *path);`

改变目录, 与用户通过`cd`命令改变目录一样, 程序也可以通过`chdir`来改变目录, 这样使得 `fopen()`, `opendir()`, 这里需要路径的系统调用, 可以使用相对于当前目录的相对路径打开文件(目录)。

- `closedir()`

`int closedir(DIR*)`

关闭目录流

## 四、程序结构

```
#include <unistd.h>
#include <sys/stat.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <dirent.h>

void printdir(char *dir, int depth) {
    DIR *dp;
    struct dirent *entry;
    struct stat statbuf;
    if ((dp = 打开dir目录) 不成功) {
        打印出错信息;
        返回;
    }
    将dir设置为当前目录;
```





```
while(读到一个目录项) {  
    以该目录项的名字为参数, 调用lstat得到该目录项的相关信息;  
    if(是目录) {  
        if(目录项的名字是".." 或".")  
            跳过该目录项;  
        打印目录项的深度、目录名等信息  
        递归调用printdir, 打印子目录的信息, 其中的depth+4;  
        //如果是目录需进行递归, 那么缩进的深度就需要+4了,  
        接近一个tab的长度.  
    }  
    else 打印文件的深度、文件名等信息  
}  
返回父目录;  
关闭目录项;  
}  
  
int main(...) {  
    .....  
}
```



输入 `ls -l` 可以看到如下信息:

```
drwxr-xr-x 3 killercat killercat 4096 2007-01-11 16:27 Desktop
drwx----- 8 killercat killercat 4096 2007-01-09 14:33 Documents
drwxr-xr-x 2 killercat killercat 4096 2006-11-30 19:27 Downloads
drwx----- 4 killercat killercat 4096 2006-12-16 20:20 References
drwx----- 9 killercat killercat 4096 2007-01-11 13:34 Software
drwxr-xr-x 3 killercat killercat 4096 2006-12-11 16:39 vmware
drwx----- 6 killercat killercat 4096 2007-01-11 13:34 Workspace
```