

File I/O

part 3

한양대학교 소프트웨어학부

Dept. of Division of Computer Science
Hanyang University



Directory

- 홈(home) 디렉토리
 - 로그인 후 사용자가 처음가게 되는 디렉토리
- 현재 작업 디렉토리(current working directory)
 - 현재 일을 하고 있는 디렉토리 (상대적 경로명은 여기서 시작)
- 파일 또는 디렉토리의 pathname
 - 절대적(absolute) – 루트 디렉토리(/)에서부터 지정
 - 보기: /home/park/book
 - 상대적(relative) – 현재 작업 디렉토리(.)에서 부터 지정
 - 보기: ./park/book

Directory (cont'd)

- 디렉토리는 그 디렉토리 안에 있는 파일 또는 디렉토리의 inode 번호와 파일 또는 디렉토리의 이름의 리스트
- inode 번호는 주어진 파일 또는 디렉토리의 stat 정보와 데이터 블록의 주소를 가지고 있다
- 모든 디렉토리는 현재 디렉토리를 의미하는 (.)과 부모 디렉토리를 의미하는 (..)을 가지고 있다
- 디렉토리는 다른 디렉토리를 가질 수 있다

Directory Access mode

r	이 디렉토리 내에 소속된 파일 또는 디렉토리의 이름을 리스트해 볼 수 있다 (이 말은 이 디렉토리 내에 소속된 파일 또는 디렉토리의 내용을 읽을 수 있음을 의미하지는 않는다)
w	이 디렉토리 내에 새로운 파일 또는 디렉토리를 생성하거나 생성된 파일 또는 디렉토리를 제거할 수 있다 (이 말은 이 디렉토리 내에 소속된 파일 또는 디렉토리에 어떤 정보를 쓸 수 있음을 의미하지는 않는다)
x	명령어 cd 또는 시스템 호출 chdir 를 사용하여 이 디렉토리로 들어갈 수 있다 (어떤 파일을 open 하거나 수행시키기 위해서는 그 파일의 절대적 pathname 에 소속된 모든 디렉토리의 x 접근 권한이 필요하다)



System Call – mkdir/rmdir

사용법

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

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

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

```
int mkdir(const char *pathname, mode_t mode);
```

```
int rmdir(const char *pathname);
```

디렉토리의 생성 및 제거

pathname : 생성 또는 제거될 디렉토리 경로

mode : 디렉토리에 대한 접근 허가권

Return value

[성공시: 0]

[실패시: -1]

Example – mkdir

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <sys/stat.h>
4 #include <sys/types.h>
5 #include <fcntl.h>
6 #include <unistd.h>
7
8 int main(int argc, char* argv[])
9 {
10     if (argc != 2) {
11         fprintf(stderr, "Usage: ./mkdir [Directory Name]\n");
12         exit(-1);
13     }
14
15     if (mkdir(argv[1], 0755)) {
16         perror("mkdir error");
17         exit(-1);
18     }
19
20     return 0;
21 }
22
```

```
[TA3@localhost lab7]$ ls
mkdir  mkdir.c
[TA3@localhost lab7]$ ./mkdir newDirectory
[TA3@localhost lab7]$ ls
mkdir  mkdir.c  newDirectory
[TA3@localhost lab7]$
```



System Call – opendir/closedir

사용법

```
#include <sys/types.h>
#include <dirent.h>
DIR *opendir(const char *dirname);
int closedir(DIR *dirptr);
```

디렉토리 열기 및 닫기

opendir

Return value

[성공시: **DIR**유형에 대한 포인터 반환]

[실패시: **NULL**]

closedir

Return value

[성공시: **0**]

[실패시: **-1**]



System Call – readdir/rewinddir

사용법

```
#include <sys/types.h>
#include <dirent.h>
struct dirent* readdir(DIR *dirptr);
void rewinddir(DIR *dirptr);
```

디렉토리 읽기

readdir

Return value

[성공시: struct dirent*]

[실패시: NULL]

rewinddir

no return value



struct dirent의 구성

```
struct dirent {  
    long    d_ino    /* 디렉토리의 inode번호 */  
    off_t    d_off;   /* 디렉토리의 offset */  
    unsigned short d_reclen /* 디렉토리 레코드 길이 */  
    char     d_name[NAME_MAX+1]; /* 디렉토리 이름 */  
};
```

Example – readdir/rewinddir

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <dirent.h>
4 #include <sys/types.h>
5
6 int main(int argc, char* argv[])
7 {
8     DIR* dirptr;
9     struct dirent* dir;
10    int i;
11
12    if ((dirptr = opendir(argv[1])) == NULL) {
13        perror("No such a directory\n");
14        exit(-1);
15    }
16
17    for (i = 0; i < 2; i++) {
18        while(dir = readdir(dirptr)) {
19            if (dir->d_ino != 0)
20                printf("%s\n", dir->d_name);
21        }
22        rewinddir(dirptr);
23        printf("=====\n");
24    }
25    closedir(dirptr);
26    return 0;
27 }
28

```

```

[TA3@localhost lab7]$ ls
mkdir  mkdir.c  newDirectory  readdir  readdir.c
[TA3@localhost lab7]$ cd newDirectory/
[TA3@localhost newDirectory]$ ls
aaaa  bbbbbb  test.txt  test2.txt
[TA3@localhost newDirectory]$ ../readdir ../newDirectory/
test.txt
..
test2.txt
bbbbbb
.
aaaa
=====
test.txt
..
test2.txt
bbbbbb
.
aaaa
=====
[TA3@localhost newDirectory]$

```



System Call – chdir/getcwd

사용법

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

```
int chdir(const char *dirpath);
```

```
char *getcwd(char *name, size_t size);
```

chdir

현재 작업 디렉토리를 변경한다

Return value

[성공시: 0]

[실패시: -1]

getcwd

현재 작업 디렉토리 경로 이름을 찾는다

name: 디렉토리 이름을 넣을 장소

size: name[]의 크기

Return value

[성공시: name]

[실패시: NULL]

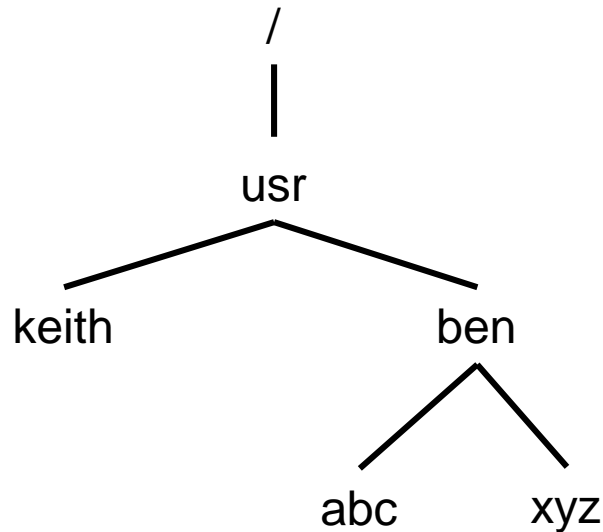


Example – chdir/getcwd

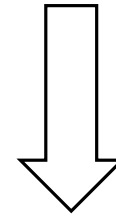
```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <unistd.h>
5
6 int main(int argc, char* argv[])
7 {
8     char buf[255];
9     getcwd(buf, 255);
10    printf("이전 작업 디렉토리 : %s\n", buf);
11
12    if (chdir(argv[1])) {
13        perror("error");
14        exit(-1);
15    }
16
17    getcwd(buf, 255);
18    printf("현재 작업 디렉토리 : %s\n", buf);
19    return 0;
20 }
21
22
```

```
[TA3@localhost lab7]$ pwd
/home/CSE4009/TA3/lab7
[TA3@localhost lab7]$ ./chdir newDirectory/
이전 작업 디렉토리 : /home/CSE4009/TA3/lab7
현재 작업 디렉토리 : /home/CSE4009/TA3/lab7/newDirectory
[TA3@localhost lab7]$ pwd
/home/CSE4009/TA3/lab7
[TA3@localhost lab7]$
```

Example – chdir/getcwd



```
fd1=open("/usr/ben/abc",O_RDONLY);  
fd2=open("/usr/ben/xyz",O_RDWR);
```



```
chdir("/usr/ben");  
fd1=open("abc",O_RDONLY);  
fd2=open("xyz",O_RDWR);
```

System Call – ftw

사용법

```
#include <ftw.h>
```

```
int ftw(const char *path, int(*func) ( ), int depth);
```

path : 열고자 하는 경로

func : 각 파일 또는 디렉토리 탐색시 불러지는 함수

int(*func) (const char *fpath, const struct stat *sb, int typeflag)

typeflag: FTW_F(일반파일), FTW_D(디렉토리)

FTW_NS(fpath 실패시)

depth : 사용할 파일 descriptor 개수(보통 1 사용)

클수록 디렉토리의 개방 횟수가 줄고 처리 속도가 빠름
한 프로세스가 할당 가능한 최대 수 존재

ftw

Return value

[성공시: 0]

[실패시: 함수 func가 반환하는 -1 혹은 0 이 아닌값]



Example – ftw(1/2)

```
1 #include <stdio.h>
2 #include <sys/stat.h>
3 #include <ftw.h>
4
5 int list(const char *name, const struct stat *status, int type)
6 {
7     if (type == FTW_NS)
8         return 0;
9     if (type == FTW_F)
10         printf("%-30s\t0%o\n", name, status->st_mode&0777);
11     else
12         printf("%-30s*\t0%o\n", name, status->st_mode&0777);
13     return 0;
14 }
15
16 int main(int argc, char* argv[]) {
17     if (argc == 1)
18         ftw(".", list, 1);
19     else
20         ftw(argv[1], list, 1);
21
22     return 0;
23 }
24
```



Example – ftw(2/2)

```
[TA3@localhost lab7]$ ./ftw
. * 0755
./chdir.c 0644
./readdir 0755
./readdir.c 0644
./mkdir 0755
./mkdir.c 0644
./ftw.c 0644
./newDirectory * 0755
./newDirectory/test.txt 0644
./newDirectory/test2.txt 0644
./newDirectory/bbbbbbb * 0755
./newDirectory/aaaa * 0755
./ftw 0755
./chdir 0755
[TA3@localhost lab7]$ ./ftw newDirectory/
newDirectory * 0755
newDirectory/test.txt 0644
newDirectory/test2.txt 0644
newDirectory/bbbbbbb * 0755
newDirectory/aaaa * 0755
[TA3@localhost lab7]$
```



과제

ftw 함수와 같이 모든 디렉토리를 탐색하고 해당 디렉토리 내에 파일 또는 디렉토리를 출력해주는 myftw를 작성하여라

(단, **ftw**를 사용하지 않고 readdir, stat함수 등을 이용!!)

1. opendir로 dir포인터 개방
2. readdir와 stat를 이용하여 파일이면 이름을 출력
3. 디렉토리이면 디렉토리 이름을 출력하고, 해당 디렉토리로 이동하여 2번 과정을 반복
4. 더 이상 파일 또는 디렉토리가 존재하지 않으면 종료

실행) \$./myftw .

현재 폴더 내에 모든 파일 및 디렉토리 이름, 하위 디렉토리 내용을 출력



제출

- 제출 방법
 - 본인 홈 계정 submit 폴더를 만들고 myftw 실행파일 생성
 - /home/학번/submit/myftw
 - 기한: 2018년 4월 13일 (금) 23시 59분 59초 (자정)
 - Email로 제출 메일 보내기
 - 이제훈: eden1948@naver.com
 - 안영재: yous4831@hamail.net

Q & A

- Thank you :)