# File I/O
## part 3

한양대학교 소프트웨어학부

*Dept. of Division of Computer Science*

*Hanyang University*

CSE4009
System Programming

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

```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <fcntl.h>
#include <unistd.h>

int main(int argc, char* argv[])
{
        if (argc != 2) {
                fprintf(stderr, "Usage: ./mkdir [Directory Name]\n");
                exit(-1);
        }

        if (mkdir(argv[1], 0755)) {
                perror("mkdir error");
                exit(-1);
        }

        return 0;
}

```

```
[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: DIR유형에 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

```c
#include <stdio.h>
#include <stdlib.h>
#include <dirent.h>
#include <sys/types.h>

int main(int argc, char* argv[])
{
        DIR* dirptr;
        struct dirent* dir;
        int i;

        if ((dirptr =opendir(argv[1])) == NULL) {
                perror("No such a directory\n");
                exit(-1);
        }

        for (i = 0; i < 2; i++) {
                while(dir = readdir(dirptr)) {
                        if (dir->d_ino != 0)
                                printf("%s\n", dir->d_name);
                }
                rewinddir(dirptr);
                printf("==================\n");
        }
        closedir(dirptr);
        return 0;
}

```

```
[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);
```

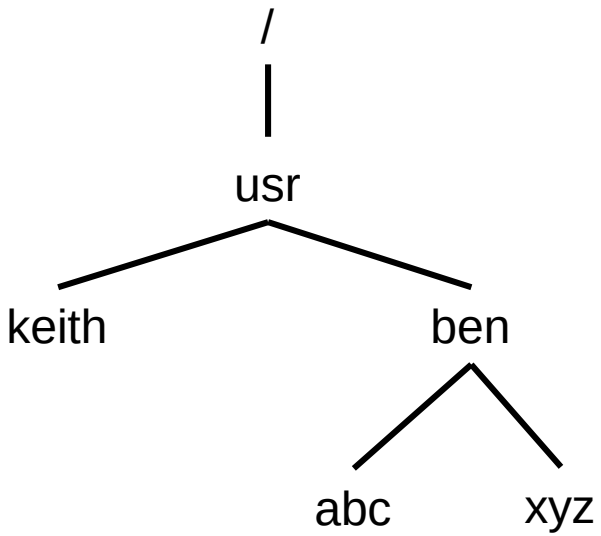
chdir
현재 작업 디렉토리를 변경한다
Return value
[성공시: 0]
[실패시: -1]
getcwd
현재 작업 디렉토리 경로 이름을 찾는다
name: 디렉토리 이름을 넣을 장소
size: name[ ]의 크기
Return value
[성공시: name]
[실패시: NULL]

# Example – chdir/getcwd

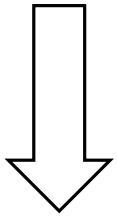
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

int main(int argc, char* argv[])
{
        char buf[255];
        getcwd(buf, 255);
        printf("이전 작업 디렉토리 : %s\n", buf);

        if (chdir(argv[1])) {
                perror("error");
                exit(-1);
        }

        getcwd(buf, 255);
        printf("현재 작업 디렉토리 : %s\n", buf);
        return 0;
}
```

```
[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 불려지는 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)

```c
#include <stdio.h>
#include <sys/stat.h>
#include <ftw.h>

int list(const char *name, const struct stat *status, int type)
{
        if (type == FTW_NS)
                return 0;
        if (type == FTW_F)
                printf("%-30s\t0%o\n", name, status->st_mode&0777);
        else
                printf("%-30s*\t0%o\n", name, status->st_mode&0777);
        return 0;
}

int main(int argc, char* argv[]) {
        if (argc == 1)
                ftw(".", list, 1);
        else
                ftw(argv[1], list, 1);

        return 0;
}
```

# 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/bbbbbb            * 0755
./newDirectory/aaaa              * 0755
./ftw                              0755
./chdir                            0755
[TA3@localhost lab7]$ ./ftw newDirectory/
newDirectory                     * 0755
newDirectory/test.txt              0644
newDirectory/test2.txt             0644
newDirectory/bbbbbb              * 0755
newDirectory/aaaa                * 0755
[TA3@localhost lab7]$
```

# 과제

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

(단, **ftw를 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 파일
  - 자기가 작성한 myftw.c 소스코드
  - 결과 화면을 캡쳐한 이미지 파일

- 제출 방법
  - 파일 압축: 학번_이름
    - Ex) 2015012789_홍길동
  - 제출기: [실습반-3] myftw
    - Ex) [307-2] myftw
  - 기한: 2017년 11월 12일 (일) 23시 59분 59초 (자정)

- 파일 압축 잘 할것. 파일 누락되면 0점

# Q & A

- Thank you :)