Part 02. 리눅스 시스템 프로그래밍

Chapter 01. 파일 I/O(1)

진행 순서

Chapter 01_01 파일 I/O 개요
Chapter 01_02 파일 생성 및 관리 (creat, access, stat)
Chapter 01_03 권한 (Permisson) (chmod)
Chapter 01_04 소유권 (Ownership) (chown)

Chapter 01_01 파일 I/O 개요

파일은 리눅스 운영체제에서 가장 기본적이고 핵심이 되는 추상화 개념입니다.
“리눅스는 모든 것이 파일이다.”

파일 디스크립터 (File Descriptor) 란?
- 파일 디스크립터는 프로세스의 열린 파일을 고유하게 식별하는 정수(int)입니다.

파일 디스크립터 테이블 (File Descriptor Table)
- 파일 디스크립터는 파일 테이블 엔트리 (File Table Entry)들을 가리키는 포인터 요소이고,
  이 파일 디스크립터를 가리키는 정수 배열의 집합을 파일 디스크립터 테이블이라 합니다.
  운영 체제에는 각 프로세스마다 하나의 고유한 파일 디스크립터 테이블이 제공됩니다.

파일 테이블 엔트리 (File Table Entry)
- 파일 테이블 엔트리는 메모리내에 존재하는 열린 파일에 대한 구조체입니다.
  이는 파일을 열거나 파일 위치를 유지 보수할 때 생성 됩니다.

표준 파일 디스크립터 (Standard File Descriptor)
- 프로세스가 시작되면 해당 프로세스 파일 디스크립터 테이블의 파일 디스크립터 0, 1, 2 가 자동으로 열립니다.
- 0 (stdin) : 키보드에서 문자를 쓸 때마다 fd 0 을 통해 stdin으로 부터 읽어 들입니다.
- 1 (stdout) : fd 1을 통해 화면의 stdout에 쓸 때마다 화면 출력에 나타납니다.
- 2 (stderr) : fd 2 를 통해 화면의 stderr에 쓸 때마다 화면에 에러 출력이 나타납니다.

Chapter 01_02 파일 생성 및 관리 (creat)

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

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

파일을 생성하기 위해 creat() 또는 open() 등 이용

creat() 함수는 이미 존재하는 파일의 경우 초기화 시키거나 존재하지 않는 파일 생성 시 사용
- open() 함수에 O_CREAT | O_WRONLY | O_TRUNC 플래그 세팅과 동일

path_name 은 생성하거나 열고자 하는 파일이름을 나타낸다.
보통 full path 이름을 적어주며, 단지 파일이름만 적을 경우에는 현재 경로에서 찾는다.
mode 인자는 파일을 생성 시 파일의 권한(소유권)을 나타낸다.

정상 수행 시 파일 디스크립터 리턴, 에러 발생 시 -1 리턴(errno)

Chapter 01_02 파일 생성 및 관리 (creat)

File mode bits:

mode
S_IRWXU : 00700 모드로 파일 소유자에게 읽기, 쓰기, 쓰기 실행권한을 준다.
S_IRUSR : 00400 으로 사용자에게 읽기 권한을 준다.
S_IWUSR : 00200 으로 사용자에게 쓰기 권한을 준다.
S_IXUSR : 00100 으로 사용자에게 실행 권한을 준다.

S_IRWXG : 00070 으로 그룹에게 읽기, 쓰기, 실행 권한을 준다.
S_IRGRP : 00040 으로 그룹에게 읽기권한을 준다.
S_IWGRP : 00020 으로 그룹에게 쓰기권한을 준다.
S_IXGRP : 00010 으로 그룹에게 실행권한을 준다.

S_IRWXO : 00007 으로 기타 사용자 에게 읽기, 쓰기, 실행 권한을 준다.
S_IROTH : 00004 으로 기타 사용자 에게 읽기 권한을 준다.
S_IWOTH : 00002 으로 기타 사용자 에게 쓰기 권한을 준다.
S_IXOTH : 00001 으로 기타 사용자 에게 실행 권한을 준다.

Chapter 01_02 파일 생성 및 관리 (creat)

creat_example.c

```
#include <stdio.h>
#include <fcntl.h> /* for mode_t */
#include <errno.h> /* for errno */
#include <string.h> /* for strerror */

int main()
{
        int fd;
        mode_t mode = S_IRUSR | S_IWUSR | S_IRGRP | S_IROTH;
        char *filename = "/tmp/file";

        fd = creat(filename, mode);
        if (fd < 0) {
                printf("creat error: %s\n", strerror(errno));
                return -1;
        }

        return 0;
}
```

```
[root@localhost ch8]# gcc -g creat_example.c -o creat_example
[root@localhost ch8]# ./creat_example
[root@localhost ch8]# ls -hl /tmp/file
-rw-r--r--. 1 root root 0  3월 30 16:46 /tmp/file
```

Chapter 01_02 파일 생성 및 관리 (access)

```
#include <unistd.h>

int access(const char *pathname, int mode);
```

access ()는 호출 프로세스가 파일 경로 이름에 액세스 할 수 있는지 확인합니다.

모드는 수행 할 접근성 검사 지정
F_OK: 파일 존재 여부
R_OK: 파일 존재 여부, 읽기 권한
W_OK: 파일 존재 여부, 쓰기 권한
X_OK: 파일 존재 여부, 실행 권한

정상 수행 시 파일 0 리턴, 에러 발생 시 -1 리턴(errno)

Chapter 01_02 파일 생성 및 관리 (access)

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

int main()
{
        char *pathname = "./creat_example";
        int mode = R_OK | W_OK;

        if (access(pathname, mode) == 0) {
                printf("Read Write OK!\n");
        } else {
                printf("You do not have permission or do not exist.\n");
        }

        return 0;
}
```

access_example.c

```
[root@localhost ch8]# gcc -g access_example.c -o access_example
[root@localhost ch8]# ./access_example
Read Write OK!
```

Chapter 01_02 파일 생성 및 관리 (stat)

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

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

리눅스는 파일의 메타 데이터 정보를 얻기위해 stat()을 제공합니다.
stat 구조체에 정보를 저장하며 stat 구조체는 <bits/stat.h>에 선언되어 있고 이는 <sys/stat.h>에 포함되어 있다.

```
struct stat {
        dev_t st_dev;       /* 파일을 포함하고 있는 장치 ID */
        ino_t st_ino;       /* inode 번호 */
        mode_t st_mode;     /* 권한 (permissions) */
        nlink_t st_nlink;   /* 하드 링크 수 */
        uid_t st_uid;       /* user ID */
        gid_t st_gid;       /* group ID */
        dev_t st_rdev;      /* device ID (특수 파일일 경우) */
        off_t st_size;      /* 바이트 단위의 전체 사이즈 */
        blksize_t st_blksize; /* filesystem I/O 를 위한 블록 사이즈 */
        blkcnt_t st_blocks; /* 할당된 블록의 개수 */
        time_t st_atime;    /* 마지막 접근 시간 */
        time_t st_mtime;    /* 마지막 편집 시간 */
        time_t st_ctime;    /* 마지막 상태 변경 시간 */
};
```

Chapter 01_02 파일 생성 및 관리 (stat)

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

int main(int argc, char *argv[])
{
        struct stat sb;
        int ret;

        if (argc < 2) {
                printf("Usage: %s <file>\n", argv[0]);
                return -1;
        }

        ret = stat(argv[1], &sb);
        if (ret) {
                perror("stat error: ");
                return -1;
        }

        printf("file(%s) is %ld bytes\n", argv[1], sb.st_size);

        return 0;
}
```

stat_example.c

파일의 사이즈 획득 예제

```
[root@localhost ch8]# gcc -g stat_example.c -o stat_example
[root@localhost ch8]# ./stat_example
Usage: ./stat_example <file>
[root@localhost ch8]# ./stat_example stat_example.c
file(stat_example.c) is 376 bytes
```

Chapter 01_02 파일 생성 및 관리 (stat)

파일의 타입 획득 예제

```
...
switch (sb.st_mode & S_IFMT) {
        case S_IFBLK:
                printf("block device node\n");
                break;
        case S_IFCHR:
                printf("character device node\n");
                break;
        case S_IFDIR:
                printf("directory\n");
                break;
        case S_IFIFO:
                printf("FIFO\n");
                break;
        case S_IFLNK:
                printf("symbolic link\n");
                break;
        case S_IFREG:
                printf("regular file\n");
                break;
        case S_IFSOCK:
                printf("socket\n");
                break;
        default:
                printf("unknown\n");
                break;
}
...
```

Chapter 01_03 권한(Permission) (chmod)

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

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

경로명이 지정된 파일의 모드를 변경합니다.

정상 수행 시 파일 0 리턴, 에러 발생 시 -1 리턴(errno)

Chapter 01_03 권한(Permission) (chmod)

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

int main()
{
        char *filename = "./creat_example";
        int mode = F_OK;

        if (access(filename, mode) == 0) {
                if (chmod(filename, S_IRWXU|S_IRWXG) != 0) {
                        printf("chmod() error\n");
                        return -1;
                }
        } else {
                printf("file(%s) access error\n", filename);
                return -1;
        }

        return 0;
}
```

chmod_example.c

```
[root@localhost ch8]# ls -hl
-rw-r--r--. 1 root root 279  3월 30 17:51 chmod_example.c
-rwxr-xr-x. 1 root root 14K  3월 30 16:46 creat_example
[root@localhost ch8]# gcc -g chmod_example.c -o chmod_example
[root@localhost ch8]# ./chmod_example
[root@localhost ch8]# ls -hl
-rwxr-xr-x. 1 root root 14K  3월 30 17:51 chmod_example
-rw-r--r--. 1 root root 279  3월 30 17:51 chmod_example.c
-rwxrwx---. 1 root root 14K  3월 30 16:46 creat_example
```

Chapter 01_04 소유권(Ownership) (chown)

```
#include <unistd.h>

int chown(const char *pathname, uid_t owner, gid_t group);
```

경로 이름으로 지정된 파일의 소유권을 변경합니다.

정상 수행 시 파일 0 리턴, 에러 발생 시 -1 리턴(errno)

Chapter 01_04 소유권(Ownership) (chown)

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

int main()
{
        char *filename = "./creat_example";
        int mode = F_OK;

        if (access(filename, mode) == 0) {
                if (chown(filename, 1, 2) != 0) {
                        printf("chown() error\n");
                        return -1;
                }
        } else {
                printf("file(%s) access error\n", filename);
                return -1;
        }

        return 0;
}
```

chown_example.c

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
...
```

/etc/passwd

```
[root@localhost ch8]# ls -hl
-rw-r--r--. 1 root root 316  3월 30 18:04 chown_example.c
-rwxrwx---. 1 root root 14K  3월 30 16:46 creat_example
[root@localhost ch8]# gcc -g chown_example.c -o chown_example
[root@localhost ch8]# ./chown_example
[root@localhost ch8]# ls -hl
-rwxr-xr-x. 1 root root   14K  3월 30 18:05 chown_example
-rw-r--r--. 1 root root   316  3월 30 18:04 chown_example.c
-rwxrwx---. 1 bin  daemon 14K  3월 30 16:46 creat_example
```