11주차 3차시 이름 있는 파이프

【학습목표】

1. 이름 있는 파이프를 설명할 수 있다.
2. 이름 있는 파이프 관련 함수를 사용할 수 있다.

학습내용1 : 이름 있는 파이프

1. FIFO란?

- 리눅스에서는 Named PIPE 기법을 위해 FIFO 라는 특수 파일을 제공.
- 부모-자식간이 아닌 독립적인 프로세스 간에 통신하기 위해서는 이름 있는 파이프 사용
- FIFO로 사용할 특수파일을 명령이나 함수로 먼저 생성해야함
- 특수 파일로 생성되는 FIFO는 생성 방법에 차이는 있지만 PIPE와 거의 유사
- 익명의 파이프는 따로 이름이 없는 통신 채널인데 비해서 FIFO 는 mkfifo() 함수에 의해 실제로 생성되는 특수 파일.
- 어떤 프로세스라도 생성된 경로로 파일에 접근하여 다른 파일을 사용하듯 읽거나 쓰기가능.
- FIFO로 pipe와 마찬가지로 단방향 통신, 반드시 읽기전용(O_RDONLY), 쓰기 전용(O_WRONLY) 으로만 열어야 함. 읽기쓰기로는 열 수 없다.

2. 명령으로 FIFO 파일 생성

; mknod나 mkfifo 명령으로 FIFO 파일 생성

① mknod 명령
- 개념 : FIFO 파일 / 특수 파일 생성
- 형식

```
mknod filename p
```

- 실행 예

```
# mknod HAN_FIFO p
# ls -l HAN_FIFO
prw-r--r--   1 root     other          0  2월 13일  12:21 HAN_FIFO
# ls -F
HAN_FIFO|
```

FIFO 표시

② mkfifo명령
- 형식 및 예

```
/usr/bin/mkfifi [-m mode] path....
```

```
# mkfifo -m 0644 BIT_FIFO
# ls -l BIT_FIFO
prw-r--r--   1 root     other          0  2월 13일  12:28 BIT_FIFO
```

3. 함수로 특수파일 생성

① 특수파일생성: mknod(2)
- 성공 시 : 0 리턴
- 실패 시 : -1 리턴

```
#include <sys/stat.h>
int mknod(const char *path, mode_t mode, dev_t dev);
```

- path : 특수 파일을 생성할 경로
- mode : 특수 파일의 종류와 접근 권한 지정
- dev : 블록/문자 장치 설정값

* mode : 생성할 특수파일의 종류 지정
- S_IFIFO : FIFO 특수 파일
- S_IFCHAR : 문자장치 특수 파일
- S_IFDIR : 디렉토리
- S_IFDIR : 블록장치 특수파일
- S_IFREG : 일반파일

② FIFO 파일 생성: mkfifo(3)
- pah로 지정한 경로에 접근 권한을 지정해 FIFO 파일 생성
- 성공 시 : 0 반환
- 실패 시 : -1 반환
- mkfifo() 함수가 정상적으로 실행되면 인자로 넘긴 경로에 특수 파일로 FIFO 가 등록됩니다.
- 다른 프로세스는 일반 파일에서 사용하는 것처럼 FIFO를 사용 가능하지만 특정 프로세스가 FIFO를 쓰기용으로 열기 전까지는 다른 읽기용 개방은 블럭(Block) 됩니다.

```
#include <sys/tpyes.h>
#include <sys/stat.h>
int mkfifo(const char *pathname, mode_t mode)
```

- const char *pathname : fifo 파일이 생성될 경로를 지정한다. 일반 파일의 경로를 지정하는 것과 동일
- mode_t mode : 생성 될 fifo 파일의 접근 권한을 지정한다.

학습내용2 : 이름 있는 파이프 관련 함수

1. FIFO로 데이터 주고받기

① 개념

- FIFO 파일을 생성하면 저수준 파일 입출력 함수로 파일을 읽거나 쓸 수 있다.
- 저수준 파일 입출력

	저수준 파일 입출력
파일 지시자	int fd;
파일 참조	파일 기술자 (file descriptor)
주요 함수	open, close, read, write, dup, dup2,

② 예제 (서버-클라이언트)

; 서버 프로세스에서 보낸 문자열을 클라이언트 프로세스가 받아서 처리하는 예

* 서버프로그램

```
...
08  int main(void) {
09      int pd, n;
10      char msg[] = "Hello, FIFO";
11
12      printf("Server =====\n");
13
14      if (mkfifo("./HAN-FIFO", 0666) == -1) {
15          perror("mkfifo");
16          exit(1);
17      }
18
19      if ((pd = open("./HAN-FIFO", O_WRONLY)) == -1) {
20          perror("open");
21          exit(1);
22      }
23
24      printf("To Client : %s\n", msg);
25
26      n = write(pd, msg, strlen(msg)+1);
27      if (n == -1) {
28          perror("write");
29          exit(1);
30      }
31      close(pd);
32
33      return 0;
34  }
```

FIFO 파일 생성

FIFO 파일 쓰기모드로 열기

FIFO 파일에 문자열 출력

* 클라이언트 프로그램

```
...
08  int main(void) {
07      int pd, n;
08      char inmsg[80];
09
10      if ((pd = open("./HAN-FIFO", O_RDONLY)) == -1) {
11          perror("open");
12          exit(1);
13      }
14
15      printf("Client =====\n");
16      write(1, "From Server :", 13);
17
18      while ((n=read(pd, inmsg, 80)) > 0)
19          write(1, inmsg, n);
20
21      if (n == -1) {
22          perror("read");
23          exit(1);
24      }
25
26      write(1, "\n", 1);
27      close(pd);
28
29      return 0;
30  }
```

서버측에서 생성한 FIFO 파일열기

서버가 보낸 데이터 읽기

```
# ex9_7s.out
Server =====
To Client : Hello, FIFO
#
```

```
# ex9_7c.out
Client =====
From Server :Hello, FIFO
#
```

【학습정리】

1. 이름 있는 파이프
- 부모-자식 프로세스 관계가 아닌 독립적인 프로세스간의 통신이 가능한 파이프
- 특수 파일의 한 종류, FIFO(First-in First-out)라고도 함
- FIFO는 명령 혹은 프로그램 안에서 함수로 생성.

2. 이름 있는 파이프 관련 함수

기능	함수원형
FIFO 생성 명령	mknod name p mkfifo [-m mode] path,,,
FIFO 생성 함수	int mknod(const char *paht, mode_t, dev_t dev); int mkfifo(const char *path, mode_t mode);