File

descriptor

2022764048 최예진

File descriptor?

프로세스에서 열린 파일의 목록을 관리하는 테이블의 인덱스

리눅스 운영 체계에서 파일에 대한 정보를 기술하고 유지하는
기억 장치의 영역

기본적으로 할당되는 파일 디스크립터?

표준 스트림

File Desciptor	Name	unistd.h	stdio.h
0	Standard input	STDIN_FILENO	stdin
1	Standard output	STDOUT_FILENO	stdout
2	Standard error	STDERR_FILENO	stderr

표준 입력 (0)

프로그램으로 들어가는 데이터 스트림이다.

프로그램은 read 명령을 이용하여 데이터 전송을 요청한다.

표준 출력 (1)

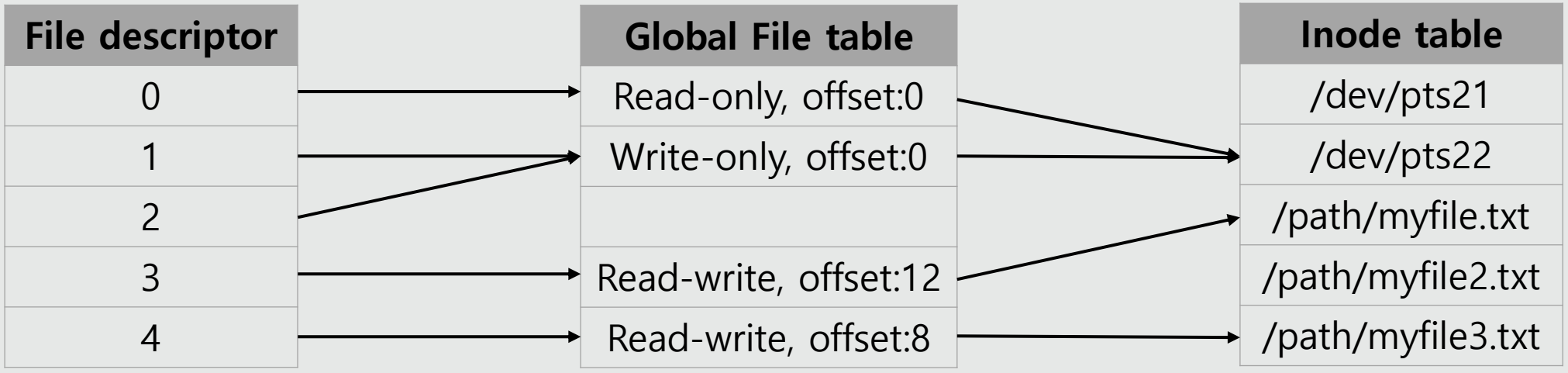
프로그램이 출력 데이터를 기록하는 스트림이다.

write 명령을 이용하여 데이터 전송을 요청한다.

표준 에러 (2)

프로그램이 오류 메시지를 출력하기 위해 일반적으로 쓰는 또다른 출력 스트림이다.

File descriptor



File descriptor

File descriptor Flag와 File Table Pointer를 가지고 있다.

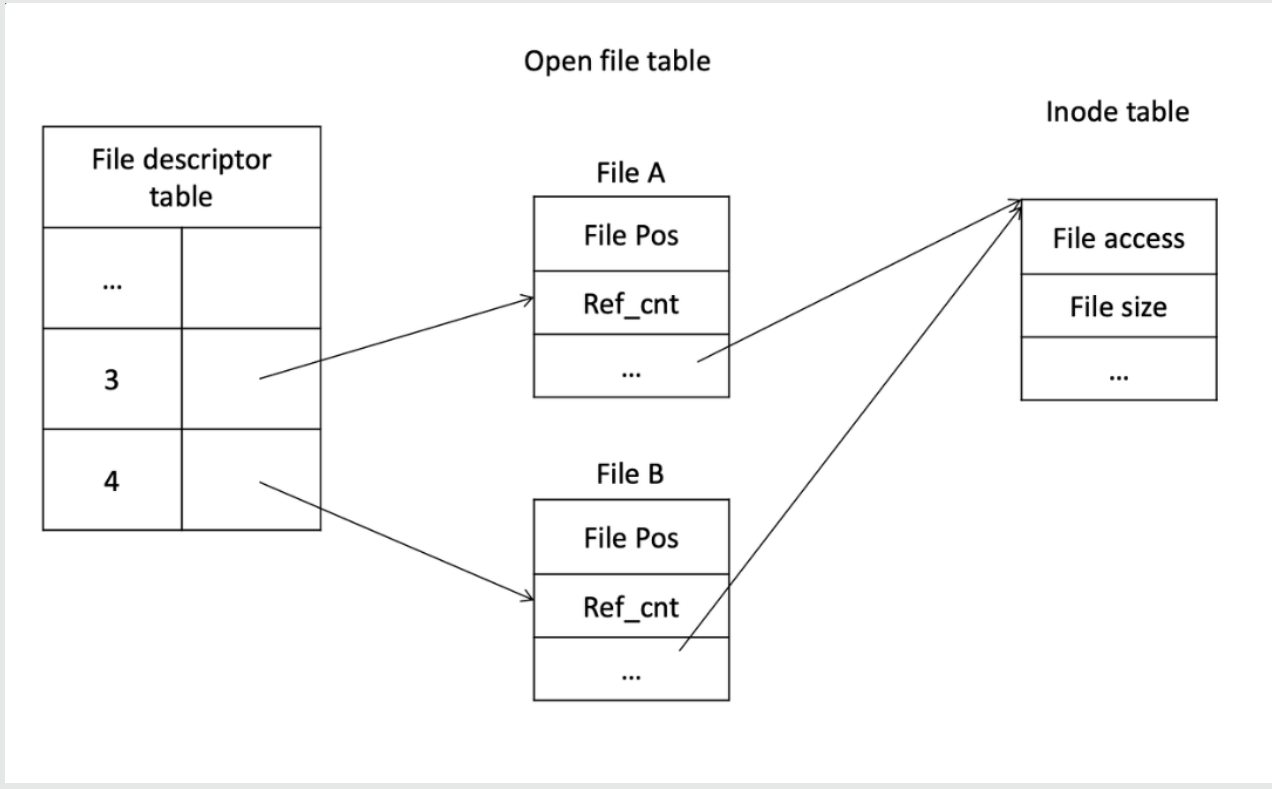
File table

mode와 inode Table Pointer의 Offset을 가지고 있다.

Inode table

해당 파일에 관한 정보를 가지고 있다.

File table



File table 정보

File descriptor table – 파일 구조체를 가리키는 포인터

File - 파일의 위치, ref_cnt 변수

Inode table - 파일의 위치를 가리키는 포인터, 파일의 정보

배경과 특징

배경

유닉스 이전의 대부분 운영 체제에서는 프로그램은 명시적으로 적절한 입력 장치와 출력 장치에 연결해줄 필요가 있었다.

하지만, 파일 디스크립터를 통해 프로그램은 어떤 장치와 연결되는지 명시적으로 알 필요 없이 입출력 장치를 연결하기 위한 그 어떤 추가 작업도 필요하지 않게 되었다.

특징

자식 프로세스는 부모 프로세스의 표준 스트림을 상속 받는다.

유닉스 시스템에서 파일 디스크립터는 정수로 표현된다.

각 프로세스는 열린 파일과 소켓에 대한 정보를 저장하는 파일 디스크립터 테이블을 가지고 있다.

네트워크 통신, 파이프, 소켓 등 다양한 입출력 작업에 사용된다.

실습

파일 디스크립터 확인

file1.c 파일과 test.txt 파일을 생성한 후 프로세스가 파일을 열 때 얻는 파일 디스크립터를 출력해보자.
test.txt파일은 아무 내용이나 넣어도 상관 없다.

dpwls@BOOK-PMTSHF3Q4P:

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

int main(void)
{
        int fd;

        fd = open("test.txt", O_RDONLY);
        if (fd < 1)
        {
                printf("open() error");
                exit(1);
        }
        printf("FD : %d\n", fd);
        close(fd);
        return (0);
}
```

file1.c 내용

dpwls@BOOK-PMTSHF3Q4P:

```
시스템 프로그래밍
~
~
~
~
```

test.txt 내용

dpwls@BOOK-PMTSHF3Q4P:

```
dpwls@BOOK-PMTSHF3Q4P:~$ mkdir test
dpwls@BOOK-PMTSHF3Q4P:~$ cd test
dpwls@BOOK-PMTSHF3Q4P:~/test$ vi file1.c
dpwls@BOOK-PMTSHF3Q4P:~/test$ vi test.txt
dpwls@BOOK-PMTSHF3Q4P:~/test$ make file1
cc     file1.c   -o file1
dpwls@BOOK-PMTSHF3Q4P:~/test$ ./file1
FD : 3
```

실행 결과

실습

기존 파일 디스크립터 복제

파일 디스크립터를 변경시킬 수 있는지 확인하기 위해서 기존에 사용 중인
파일 디스크립터 번호를 복제해 보자.

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

int main(void)
{
        int fd;
        int fd2;

        fd = open("test.txt", O_RDONLY);
        fd2 = open("test.txt", O_RDONLY);
        if (fd < 1 || fd2 < 1)
        {
                printf("open() error");
                exit(1);
        }
        printf("fd\t: %d\n", fd);
        printf("fd2\t: %d\n", fd2);

        printf("fd2 = dup(fd)\n");
        fd2 = dup(fd);

        printf("fd\t: %d\n", fd);
        printf("fd2\t: %d\n", fd2);

        close(fd);
        close(fd2);
        return (0);
}
```

file2.c 내용

```
dpwls@BOOK-PMTSHF3Q4P:~/test$ vi file2.c
dpwls@BOOK-PMTSHF3Q4P:~/test$ make file2
cc     file2.c   -o file2
dpwls@BOOK-PMTSHF3Q4P:~/test$ ./file2
fd      : 3
fd2     : 4
fd2 = dup(fd)
fd      : 3
fd2     : 5
```

실행 결과

Reference

https://velog.io/@hyeseong-dev/File-Descriptors
https://code4human.tistory.com/123
https://dev-ahn.tistory.com/m/96?amp=1
https://sikpang.tistory.com/19
https://codable.tistory.com/19
https://en.wikipedia.org/wiki/File_descriptor

https://80000coding.oopy.io/cdc3317b-8077-4a35-8cd6-1f493aa67ef9 - 실습참고

감사

합니다