

## 용어설명

### 1. 탐색기 . shell . file system . bit . byte . 2bit . 4bit . 디렉토리 . 폴더

- 1) 탐색기 : 파일이나 폴더를 찾을 수 있도록 하는 프로그램
- 2) Shell : Unix . Linux 입장에서 shell은 운영체제와 U/R 사이 계층 역할을 하는 명령어 해석기 이면서, 변수를 정의할 수 있는 조건문, 반복문, 산술연산 등을 할 수 있는 프로그래밍 언어이다.
- 3) file system : 파일의 이름을 지정하고 저장, 검색을 위해 어디에 저장시켜야 하는지에 대한 방법을 규정한 시스템
- 4) bit : 데이터의 가장 작은 단위로 1bit는 0.1 둘 중 1개 값을 가질 수 있다 (binary digit)
- 5) byte : (8bit) 데이터 파일의 크기, 메모리를 통해 전송되는 데이터 양을 표현 하는 데 사용됨
- 6) 32 bit : 컴퓨터 구조에서 32bit 정수, 메모리 주소, 다른 데이터 장치들은 32bit 영역을 점 32bit는 중앙처리 장치 주소가 32bit 단위씩 직렬로 전송하는 컴퓨터 세대를 가리키고하여 32bit 컴퓨터라 한다.
- 7) 64 bit : 컴퓨터 구조에서 64bit 정수, 메모리 주소, 다른 데이터 장치들은 64bit 영역을 점 64bit는 중앙처리 장치 주소가 64bit 단위씩 직렬로 전송하는 컴퓨터 세대를 가리키고하여 64bit 컴퓨터라 한다. 32bit 이하의 연산 처리 속도가 낮아지며 최근엔 64bit 컴퓨터가 대량이다.

## 2. 디렉토리 . 폴더

- 1) 디렉토리 : 파일 시스템의 관점에서 파일들을 묶어두는 개념.
- 2) 폴더 : 디렉토리보다 포괄적 개념으로 파일시스템에서 존재하지 않는 특권층을 가지는 이름을 쓸 수 있는 개념.

## 3. 파일명규칙 (리눅스 기준)

- 1) 기본적으로 "/"를 제외한 어떤 ASCII 문자도 파일명으로 사용 가능
- 2) 길이는 255자를 제한. 대문자 소문자
- 3) 확장자는 큰 의미가 없으나 컴파일러나 특정 프로그램에서 확장자를 인식하여 다른 동작을 하는 경우도 있음.
- 4) 대부분의 Linux 프로그램을 backup파일 이름붙이기 - , ~를 치니.
- 5) "."으로 시작되면 보통 Hidden Name.

## 4. Window file system . Unix file system . Apple file system .

- 1) Window - FAT (File Allocation Table) : Dos 때부터 사용하고 있는 대표적인 파일시스템 하드디스크에 FAT 영역을 만들어 하드디스크에 저장되어있는 파일의 실제 위치 등의 정보를 기록하여 이를 이용하는 시스템  
- NTFS : Master File Table을 사용하여 디렉터리 구조와 파일의 유대관계를 시스템이 복귀하는
- 2) Unix : 유닉스 개념으로 모든 파일들은 byte 단위의 스트림으로 간주
- 3) Apple : 애플에서 mac OS, watch OS 등 다양한 운영체제에 사용되는 것으로, 기존 HFS+를 대체한 (APFS)

## 정리하기

1. 디렉토리란 현재 작업중인 위치를 나타낸다.  
리눅스는 ./ 03 현재 디렉토리 ../3 현재 디렉토리에 대한 상위 디렉토리를 표현한다  
이 표시에 대한 한 함성 현재 디렉토리에 대한 상대적 위치 개념으로 ./나 ../를 사용, 혹은 바로 상위 하위 디렉토리를 갈때 해당 디렉토리명만 cd 명령어 뒤에 붙이는 것을 상대적 경로라 한다.  
반면 절대 경로의 경우 현재 디렉토리를 홈 디렉토리 내의 'kop08'이라 한다면 /home/kop08 이라는 생략없이 정확하게 나타낸다.  
(./././ 현재 디렉토리 : kop08 하위 디렉토리 상대 경로)
2. 1) pwd (print name of current / working directory) : 현재 작업중인 디렉토리를 절대경로로 출력  
2) cd (change directory) : 작업중인 디렉토리 변경  
3) ls (list) -a : '.'을 포함한 디렉토리 내 모든 파일 출력  
-l : 해당 디렉토리의 정보를 자세히 표현 (이름, 용량, 권한)  
-n : '-l'과 비슷하지만 사용자나 그룹정보를 UID와 GID로 표현
3. cat filename - file 내용을 그대로 보여줌  
cat filename more - 화면단위로 보여줌  
| less - 현재까지 보여준 용량일 수 있음  
head -n filename - 첫 n줄을 출력  
tail -n filename - 마지막 n줄을 출력  
tail -f filename - 특정종인 파일의 변동 여부를 실시간 확인
4. mv filename1 filename2 file의 이름을 file2로 변경 (이름만변화)  
filename path filename path file을 해당 경로 옮김  
dirname1 dirname2 디렉토리 디렉토리 변경  
cp filename1 filename2 file1을 file2에 복사 (내용변도)  
file1과 file2가 존재하는 디렉토리 생성하여 복사함  
filename path file을 해당 경로 복사  
rm filename file 삭제  
rmdir dirname directory 삭제 (단, 디렉토리가 비워져 있어야함)  
rm -d 빈 directory를 지움  
rm -i 사용자에게 삭제할 문맥을 설명한후 명령어 실행  
rm -rf 현재 디렉토리 모든파일은 삭제하여 모두 삭제
5. history - 명령어 히스토리 불러내기  
! 숫자 ~번째 실행된 명령어 재실행  
! 키워드 해당 키워드 패턴을 가진 가장 최근 명령어 재실행  
r 재실행 (Unix에서 사용)
6. mkdir 디렉토리 만들기  
rm dir 빈 디렉토리 삭제  
rm -rf dirname 해당 디렉토리의 전체 삭제
7. wc (word count) 줄(line)과 단어(word)와 글자(character) 수 출력 가능

## ● 3. 이해하기

### (1) 파일과 디렉토리 다루기

리눅스 서버 환경은 윈도우에서 파일 탐색기를 사용하는 것과는 다르게 GUI(Graphic User Interface) 환경이 아니며 하나 하나 명령어 쉘 상태에서 파일과 디렉토리 관련 명령어를 입력하여야 한다. 이러한 명령어를 하나하나 배워보도록 한다.



## ● 3. 이해하기

### 1) 디렉토리, 절대경로

윈도우 OS의 파일을 폴더의 모아두는 개념과 마찬가지로 리눅스/유닉스에서는 디렉토리와 파일의 개념을 사용한다

① 디렉토리의 위치는 절대경로와 상대경로로 표시됨

② 윈도우에서 최상위 경로는 하드디스크 등 디바이스의 최상위 디렉토리(폴더)로 예를 들어 “C:\W”로 표시되며 이때 최 상위 경로는 슬래쉬 기호“/”, “W”로 표시된다. 유닉스와 리눅스에서는 슬래쉬가 아닌 역 슬래쉬 “\” 기호가 최상위 디렉토리이며, 루트(root) 디렉토리라고 읽는다.

③ 절대경로

· 루트(root) 디렉토리부터 현재 파일이 위치한 디렉토리의 경로를 전체로 표기한 경로

· 예 : /home/kopocte/test/help.txt

④ 상대경로 **생략되어 있는 경로는 상위에서**

· 명령어 쉘 상태에서 현재 위치로부터 파일이 있는 디렉토리를 표기한 경로

· 현재 위치를 점으로 나타내서 표기

· 현재 위치는 점을 한 개“.”, 상위디렉토리는 점이 두 개“..”로 표시

Tip  
5)

예 : 현재 위치가 /home/kopocte 라면

./test/help.txt (상대경로 표시) -> /home/kopocte/test/help.txt

../home/kopocte/test/help.txt : ..은 상위 디렉토리를 의미하며, 결국 /home/kopocte/test/help.txt 과 같은 위치

## ● 3. 이해하기

### 3) 파일내용보기

텍스트(TEXT)파일의 내용을 볼 수 있는 명령으로 cat, page, head, tail이 있으며, 파일의 내용이 많을 경우 화면 단위로 끊어서 보기위한 명령어로 more가 있다.

#### ①cat

·cat filename : filename이라는 파일의 내용을 한 번의 출력 <그림 136>참조

#### ②more

·(파이프)를 같이 사용하여 화면 단위로 출력 *box형식으로 출력*

·예: "cat filename | more"

#### ③less

·cat filename | less : filename출력하고 이동할 수 있음.

·멈춰진 현재 화면에서 방향키는 누르면 이동할 수 있음.

#### ④head

·파일의 내용을 맨 앞을 기준으로 보여줌

·head -n filename : filename의 파일을 처음 n줄을 표시

·예: head -10 abc.txt : abc.txt파일을 처음부터 10줄을 보여줌

#### ⑤tail

·파일의 내용을 맨 뒤를 기준으로 보여줌

·tail -n filename : filename의 파일을 뒤부터 n줄을 표시 *ctrl + C*



## ● 4. 실습하기(1)

### 1) 파일탐색

① [pwd]

② [cd]

③ 상위 디렉토리, 하위 디렉토리

④ 절대경로, 상대 경로

⑤ ./ .. \* : 특별한 문자의 사용

⑥ [ls -adlsFR]

### 2) 파일 내용보기

① [cat filename]

② "cat filename | more"

③ [cat filename | less]

④ [head -n filename] *처음 n줄*

⑤ [tail -n filename] *끝에서 n줄*

⑥ [tail -f filename]

### 3. 이해하기

#### 4) 파일 다루기

파일을 복사하고 파일 이름을 바꾸는 명령으로 mv, cp가 있다. <그림 137>

##### ①mv

·파일을 이동 (잘라내서 복사)하는 명령

·mv oldfilename newfilename : oldfile의 파일을 newfile로 이동함. 결국 파일명을 바꾸는 명령 (예: mva.txt b.txt : a.txt파일명을 b.txt로 바꿈)

·mv filename dirname : 파일을 해당 디렉토리로 보냄. 뒤의 인자가 파일명이 아니라 디렉토리명인 경우 해당 디렉토리 에 파일을 옮기는 결과가 됨 (예: mva.txt adir : a.txt파일을 adir의 디렉토리로 옮기는 명령이며 결국 mva.txt adir/a.txt와 같은 명령으로 뒤 인자의 a.txt가 생략된 것으로 인식된 것)

·mv oldDirName newDirName : 두인자의 명칭이 디렉토리인 경우, oldDirName의 모든 파일을 newDirName이라는 디렉토리를 만든 후 옮기는 명령이 됨. 결국 결과는 디렉토리 명을 바꾸는 명령임

##### ②cp

·파일을 복사(기존파일은 남음)하는 명령

·cp oldfilename newfilename : oldfile의 파일을 newfile로 복사함.

##### Tip

cp a.txt b.txt : a.txt파일을 b.txt로 복사하는 명령으로 실행후 a.txt와 b.txt파일 두 개가 존재

·cp filename dirname : 파일을 해당 디렉토리로 보냄. 뒤의 인자가 파일명이 아니라 디렉토리명인 경우 해당 디렉토리 에 파일을 복사하는 결과가 됨

### 3. 이해하기

##### Tip

cp a.txt a : a.txt파일을 a의 디렉토리로 복사하는 명령이며  
adir/a.txt : a.txt파일을 adir의 디렉토리로 복사하는 명령으로 뒤 인자의 a.txt가 생략된 것으로 인식된 것

·cp oldDirName newDirName : 두인자의 명칭이 디렉토리인 경우, oldDirName의 모든 파일을 newDirName이라는 디렉토리를 만든 후 복사하는 명령이 됨. 작동하지 않음.

·실제는 cp -R oldDirName newDirName 명령이 많이 사용됨. -R 옵션은 하위디렉토리까지 모두 복사하는 명령

##### ③rm

·파일을 지우는 명령

·rm filename: 파일을 지움 (예: rm a.txt a.txt파일을 지움)

·rm -i filename:파일을 지움, 단 확인 메시지가 나와서 지울것인지 묻고 yes를 선택시만 지움

·rm -r : 해당 위치의 파일 및 디렉토리를 지움 (그리고 하위 디렉토리도 지움)

·rm -d : 빈 디렉토리를 지움

·rm -f : 파일을 지울 때 확인 물음을 하지 말고 강제로 지움

·다음 명령어 사용은 주의하여야함 [m \*][m -f][m -rf]

·rm \* : 모든 파일을 다 지우는 명령

·rm -rf : 해당 위치의 모든 파일 및 디렉토리를 묻지도 말고 무조건 지움. 만일 최상위 루트(root)디렉토리에서 해당 명령을 실행시 서버의 모든 파일이 포맷(format)수준으로 지워짐. 회사 운영자가 이 명령을 잘못 사용하다간 사표쓰는 경우가 생김.

## ● 4. 실습하기(2)

### 3) 파일 다루기

- ① `[mv /oldfilename newfilename]`  
이름 변경 : 선택
- ② `[mv filename dirName]`
- ③ `[mv oldDirName newDirName]`
- ④ `[cp filename {path/}filename]`
- ⑤ `[cp filename path]`
- ⑥ `[cp DirName newDirName]`

### 4) 파일 지우기

- ① `[rm filename]`
- ② `[rm -i filename]`

주의 `[rm *]` `[rm -f]` `[rm -rf]`

## ● 3. 이해하기

### 5) 명령어 히스토리

리눅스 명령은 윈도우 시스템과 달리 매번 명령어를 입력하기 때문에 불편할 수 있다. 하지만 이전에 입력한 명령어를 다 시 찾아 명령할 수 있는 기능이 있는데, 이 명령어 히스토리 기능을 이용하면 명령어 사용이 편리해 진다. <그림 1-38>

①[r 또는 !]

·유닉스 계열 (korn shell)에서는 ‘!’를 입력하고 리눅스 계열(c shell)에서는 ‘!’(느낌표)를 입력하면 기존 사용했던 명령어를 찾을 수 있음

·예를 들어 ‘!’이라고 명령하면 기존 사용한 명령어 중 c로 시작하는 명령을 실행한다

②[화살표 위,아래]

·지금까지 사용한 명령어를 순차적으로 보여줌.

```
root@kopoctc:/home/kopoctc# ls
help2.txt  help3.txt
root@kopoctc:/home/kopoctc# !c cp help.txt help2.txt
cp: cannot stat 'help.txt': No such file or directory root@kopoctc:/home/kopoctc#
```

<그림 1-38> 명령어 히스토리 사용  
명령어 히스토리를 위해 입력되는  
logout 시점에 반영되는

③명령어 히스토리 저장 파일

·지금까지 사용한 명령어는 해당 사용자의 기본 디렉토리 내에 ‘bash\_history’(리눅스 bash) 또는, ‘history’(일반적 유닉스)라는 파일안에 저장되어 있음. <그림 1-39>

### ● 3. 이해하기

#### 7)파일의 문자수 세기

파일의 글자 수를 세는 명령어로 wc가 있는데, 일반적으로파일의 크기를 글자 수로 알고 싶을 때 나, 쉘 프로그래밍에서 응용하여 많이 사용된다. <그림 141>

①wc

·파일 내부의 글자 수 및 줄 수를 보여줌

·[wc] : 출력되는 순서는 파일의 줄 수[newline], 단어 수[word], 글자 수[byte]로 보여줌

vi 에서  
; %! %d %d %d byte 수 확인 가능

```
kopoctc@kopoctc:~/mydir$ ls -al
total 12
drwxrwxr-x 2 kopoctc kopoctc 4096 Jan 18 10:56 .
drwxr-xr-x 4 kopoctc kopoctc 4096 Jan 18 10:55 ..
-rw-rw-r-- 1 kopoctc kopoctc  27 Jan 18 10:55 myfile
kopoctc@kopoctc:~/mydir$ cat
myfile
You are a goodguy Me too.
kopoctc@kopoctc:~/mydir$ wc myfile
 2  7 27 myfile
```

<그림 1-41> wc 명령어사용