Linux System Administration

Using VI Editor

Linux Text Editors

- Text Editor
  - 텍스트 에디터는 문자 기반으로 파일을 생성하고 수정할 수 있는 유틸리티
  - 프로그래밍에서는 소스 코드의 입력과 수정이 가장 필수적인 작업
  - 유닉스에서는 다음과 같은 에디터를 지원
- ed/ex
  - 메모리 소모가 적고 속도는 빠르지만 라인 단위로 편집하기 때문에 다소 불편
- vi (vim)
  - 대부분 유닉스에서 지원하는 에디터. 강력한 기능을 제공하지만 기능이 너무 많은 것이 단점
  - 리눅스에서는 보다 편리한 사용법으로 성능을 개선한 vim(iMproved)을 사용
- GNU nano
  - 유닉스에서 지원하는 에디터로 vi 보다 편리한 편집 기능을 이용
- Emacs
  - GNU를 만든 리처드 스톨만이 개발한 에디터. 기본 제공되지 않기 때문에 별도 설치 필요

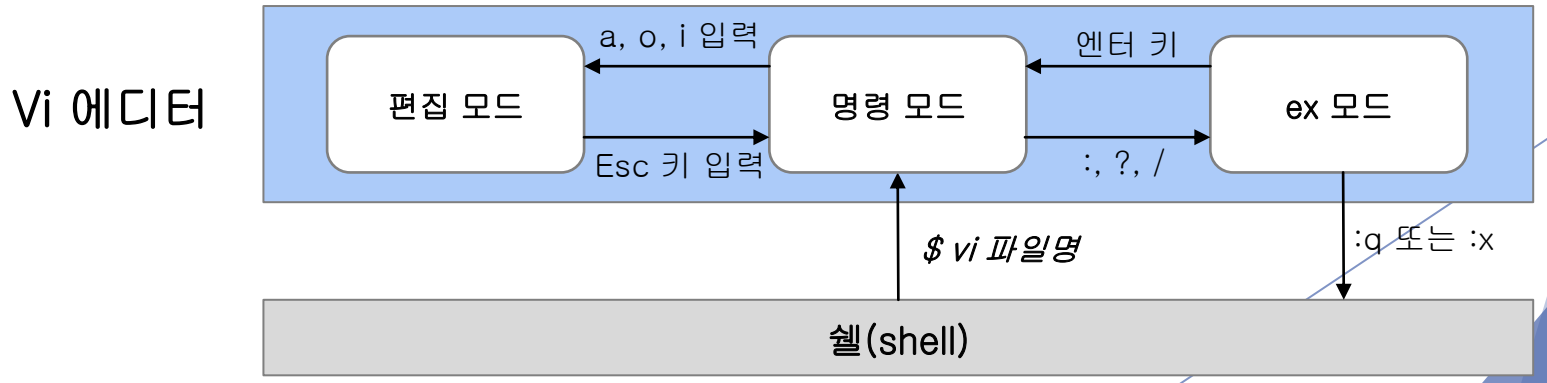
vi Editor

- vi 에디터 시작

$ vi 파일명

  - 지정한 파일명을 가진 파일이 없으면 새 파일, 있으면 기존의 파일을 엶

- vi 에디터 모드
  - 명령 (Command) 모드
  - 편집 (Edit) 모드
  - ex 모드



vi Editor
: 입력/편집 모드

- 다음 명령키로 명령 모드에서 입력/편집 모드로 전환, 편집 작업을 시작

명령키	수행 작업
i	커서 앞에 삽입
a	커서 뒤에 삽입
o	현재 줄 다음에 삽입
I	현재 줄 첫 칸 앞에 텍스트 입력
A	현재 줄 끝에 텍스트 입력
O	현재 줄 앞에 삽입

vi Editor
: 저장 및 종료 (명령 모드)

- 저장

명령키	수행 작업
:w ↵	현재의 파일명으로 파일 저장
:w 파일명 ↵	지정한 파일명으로 파일 저장

- 종료

명령키	수행 작업
:q ↵	작업 내용을 저장하였으면 vi 종료
:q! ↵	작업내용을 저장하지 않고 vi 종료
:wq↵	작업 내용을 저장한 후 vi 종료
:wq 파일명↵	작업 내용을 지정한 파일명으로 저장한 후 vi 종료
zz (shift-zz)↵	작업 내용을 저장한 후 vi 종료

vi Editor
: [실습]

- 실습 디렉터리 만들기
  - 사용자 home 디렉터리에 docs 디렉터리를 만들고
  - docs 디렉터리 안에 test.txt 라는 이름의 파일을 vi 에디터로 열어봅시다
- 문서의 편집
  - vi로 열어둔 test.txt 안에 다음과 같은 내용을 입력하여 저장해 봅시다

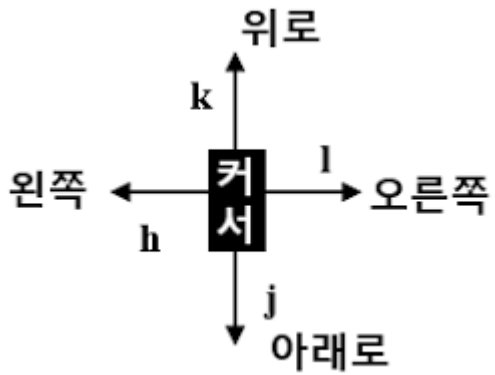
Name: {자신의 이름}
Class: Linux System Administration
Edit with: VI Text Editor

Linux is a family of free and open-source software operating system built around the Linux kernel.

GNU is a recursive acronym for "GNU is Not Unix!"

vi Editor
: 커서의 이동

- 편집 모드에서는 커서를 움직일 수 없음
- 편집 모드에서 커서를 움직이기 위해
  명령 모드로 전환하고 다음 키 조합으로 커서를 이동
- vim에서는 화살표 키로 이동 가능
- 기본 이동: h, j, k, l



이동	명령어
한 행 위	k
한 행 아래	j
한 문자 오른 쪽	l
한 문자 왼쪽	h
행의 시작	^ 또는 0
행의 마지막	$
이전 행의 처음	-
다음 행의 처음	+ 또는 ↵

vi Editor
: 커서의 이동

- 지정한 곳으로 이동하기 위한 명령

이동	명령키
줄 번호 n 위치로	:n 또는 nG
파일의 끝 줄로 이동	:$ 또는 G
n줄 만큼 앞으로 이동	n+
n줄 만큼 뒤로	n-
현재 문장의 처음으로	(
다음 문장의 처음으로	)
현재 문단의 처음으로	{
다음 문단의 처음으로	}

vi Editor
: [실습]

- 커서 이동 연습
  - 커서를 1행으로 이동 - **1G** 또는 **:1**
  - 다음 단어로 이동 - **w**
  - 다음 행으로 이동 - **j**
  - 커서를 좌우로 이동 - **h, l**
  - 마지막으로 이동 - **G** 또는 **:$**

  - 이동 명령을 이용, 자유롭게 커서를 이동할 수 있도록 연습해 봅니다

vi Editor
: 삭제 및 취소

- 명령 모드에서 동작

명령어	삭제 대상	수행 작업
x, #x	문자	커서 위치의 문자 삭제(예:3x)
dw, #dw	단어	커서 위치의 단어 삭제
dd, #dd	줄	커서 위치의 줄 삭제
D(shift-d)	줄의 일부	커서 위치부터 줄 끝까지 삭제
u		방금 수행한 명령 취소
U		해당 줄의 모든 편집 취소

vi Editor
: 편집 - 복사/잘라내기/붙이기

- 명령 모드에서 동작

명령어	수행 작업
yy, #yy	현재 행을 버퍼로 복사 (예:4yy)
p	현재 행 다음에 버퍼 내용 삽입
P	현재 행 위쪽에 버퍼 내용을 삽입
dd, #dd	현재 행을 잘라내기

- 버퍼
  - vi는 작업 내용을 버퍼에 저장 - 실행 취소 가능함
  - 복사하기, 잘라내기에 이용

vi Editor
: 편집 – 복사/잘라내기/붙이기

- 버퍼의 종류
  - Unnamed Buffer (이름 없는 버퍼)
  - Named Buffers (이름이 있는 버퍼) : “a, “b … “z
  - Numbered Buffers (번호가 붙은 버퍼) : “1, “2 … “9

- 버퍼의 활용 예
  - “a3yy -> 현재 행부터 아래로 3줄을 a 버퍼에 저장
  - “ap -> a 버퍼의 내용을 현재 행 다음에 삽입

vi Editor
: 검색

- ex 모드에서 동작

명령어	수행 작업
/문자열	현재 위치부터 파일 앞쪽으로 문자열 탐색
?문자열	현재 위치부터 파일 뒤쪽으로 문자열 탐색
n	다음 문자열 탐색
N	역방향으로 문자열 탐색

vi Editor
: 실습

- 앞서 작성한 **test.txt** 파일을 백업해 두고
- **VI Editor**의 편집, 삭제, 검색 기능을 연습해 봅니다

vi Editor
: 기타 기능

- vi에서 쉘 명령 실행

명령어	수행 작업
:!명령	vi를 중단하고 지정한 명령 수행 (vi로 돌아올 때 : ↵)
:sh	vi를 잠시빠져나가서 쉘을 수행 (vi로 돌아올때 : exit)

vi Editor
: 기타 기능

- 알아두면 유용한 명령 집합

명령어	수 행
:f 파일명	파일 이름을 지정한 이름으로 변경
:w %.old	현재 파일을 .old 이름으로 저장해 둘 때
^g	기본적인 파일정보 출력(파일명, 라인 수 등)
J	현재 줄과 다음 줄 연결
.	바로 이전에 수행한 명령 재 실행
~	현재 커서 위치의 한 문자를 소문자 혹은 대문자로 전환