

# 5장

Linux Editor 사용하기

# 전체 내용

Linux Editor  
개요

vi /vim 사용하기

Nano 사용하기

Gedit 사용하기

# 1 – Linux Editor 개요

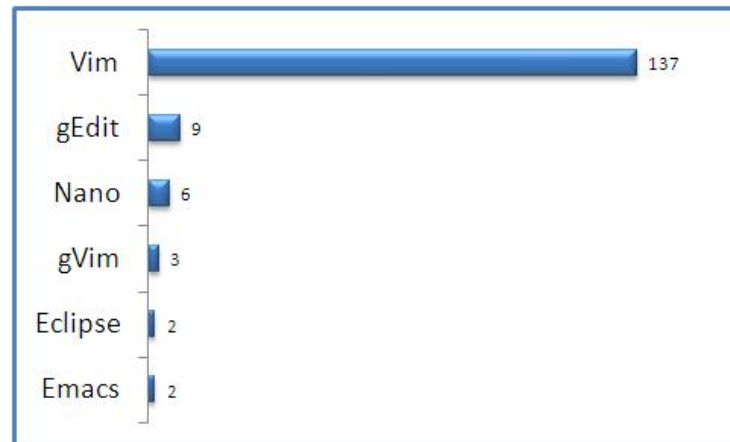
Linux Editor 종류

모드형 편집기 및 비모드형 편집기

# 1 – Linux Editor 개요

- Linux Editor 종류

- Terminal 환경에서 사용하는 대표적인 편집기는 vi/vim이다
- GUI 환경에서 주로 사용하는 것은 Gedit이다
- Linux User들이 가장 많이 사용하는 Editor의 목록이 다음과 같다



# 1 – Linux Editor 개요

- 모드형 편집기 및 비모드형 편집기
  - 모드형 편집기: vi
    - 명령어 모드: 복사, 붙여 넣기, 잘라내기, 삭제하기
    - 입력 모드: 명령어 모드 상태에서 **i**를 입력하여 내용을 입력하고 수정하기
    - 실행 모드: 입력 모드 상태에서 **esc** 그리고 **:**을 입력하여 실행모드로 들어가서 종료, 저장, 검색 및 파일 로드하기
  - 비모드형 편집기: gedit

구분		모드형(vi)	비모드형(메모장)
입력 모드		텍스트 입력	
명령 모드(사례)	복사	yy	<b>Ctrl</b> +C
	붙이기	p	<b>Ctrl</b> +V
	저장	:wq, ZZ	<b>Ctrl</b> +S
모드 전환		i, a, o, esc	해당 없음

## 2 – vi / vim 사용하기

Vi 동작 모드 이해하기

Vi 시작하고 종료하기

입력 모드로 전환하기

커서 및 화면 이동하기

내용 수정하기

내용 삭제하기

명령 취소하기

## 2 – vi / vim 사용하기

복사하기 또는 잘라서 붙여 넣기

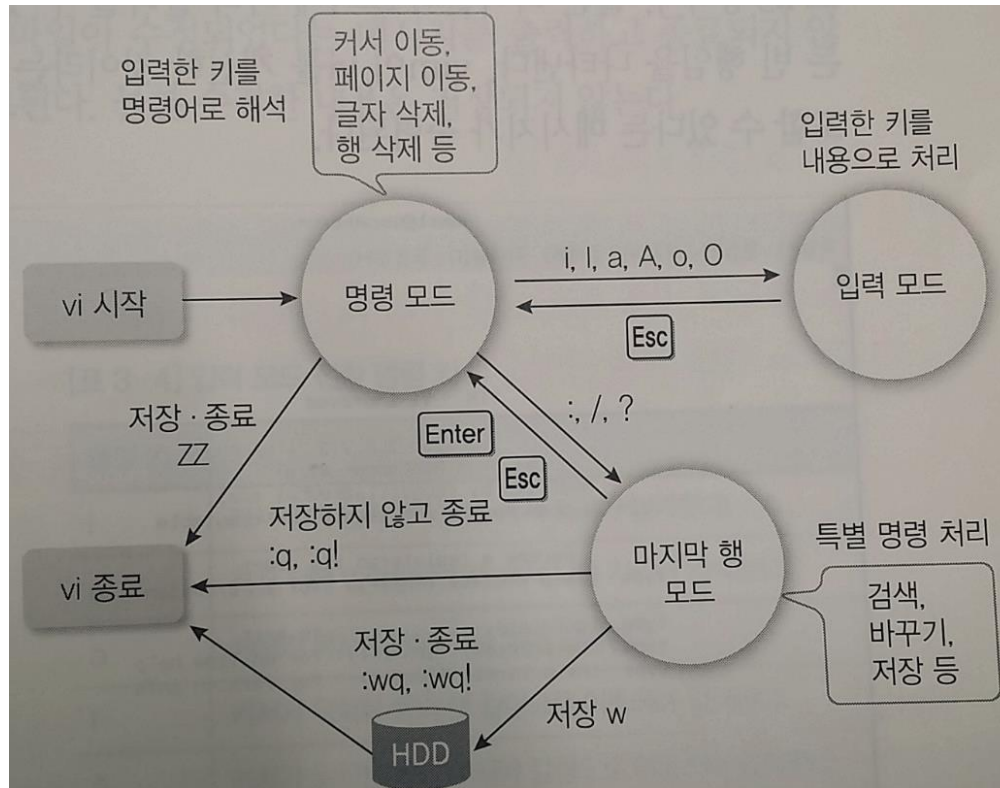
검색하기

기타 vi에서 유용한 명령 키

## 2 – vi / vim 사용하기

### • Vi 동작 모드 이해하기

- vi를 실행하면 기본적으로 [명령어 모드]로 들어 간다
- 내용을 입력하고 수정하려면 **i**를 입력하여 [입력 모드]로 들어 간다
- 편집이 끝난 후 저장하려면 **esc**를 눌러서 [실행 모드(마지막 행 모드)]로 빠져나가서 **;**, **/**, **?**중 하나를 눌러서 작업을 하고 Enter를 누른다





## 2 – vi / vim 사용하기

- Vi 시작하고 종료하기

- 시작하기

- vi test.txt
    - vi

- 파일 저장하고 종료하기

- 편집을 완료하고 vi를 종료하려면 [명령어 모드]나 [실행 모드(마지막 행 모드)]에서 명령을 입력해야 한다
    - [명령어 모드]에서 파일의 저장과 종료를 동시에 하려면 **ZZ**를 누르거나 **:wq!**을 누른다
    - [입력 모드]에서 [실행 모드]로 빠져 나가려면 esc 그리고 :를 눌러서 q,w,wq를 누른다

구분	명령 키	기능
마지막 행 모드	:q	vi에서 작업한 것이 없을 때 그냥 종료한다.
	q!	작업한 내용을 저장하지 않고 종료한다.
	:w [ 파일명 ]	작업한 내용을 저장만 한다. 파일명을 지정하면 새 파일로 저장한다.
	:wq, :wq!	작업한 내용을 저장하고 vi를 종료한다.
명령 모드	ZZ(Shift+zz)	작업한 내용을 저장하고 vi를 종료한다.

## 2 – vi / vim 사용하기

- 입력 모드로 전환하기

- Vi를 시작하면 무조건 [명령어 모드]이다. 이제 [입력 모드]로 전환하여야만 내용을 입력할 수 있다
- 이 때 사용하는 것이 i,a,o이다

- **mkdir ~/mod5**

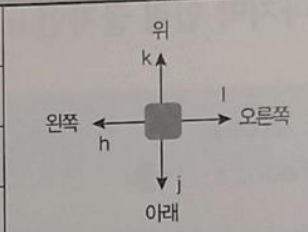

- vi test.txt**

- Welcome to Linux World*

명령 키	기능
i	커서 앞에 입력한다(현재 커서 자리에 입력한다).
a	커서 뒤에 입력한다(현재 커서 다음 자리에 입력한다).
o	커서가 위치한 행의 다음 행에 입력한다.
I	커서가 위치한 행의 첫 칼럼으로 이동하여 입력한다.
A	커서가 위치한 행의 마지막 칼럼으로 이동하여 입력한다.
O	커서가 위치한 행의 앞 행에 입력한다.

## 2 – vi / vim 사용하기

- 커서 및 화면 이동하기
  - [End], [Home], Arrow key 사용

명령 키	기능	
k	커서를 한 행 위로 이동한다.	
j	커서를 한 행 아래로 이동한다.	
l	커서를 한 글자 오른쪽으로 이동한다.	
h	커서를 한 글자 왼쪽으로 이동한다.	
^ 또는 0	커서를 현재 행의 처음으로 이동한다.	
\$	커서를 현재 행의 마지막으로 이동한다.	
-	커서를 앞 행의 처음으로 이동한다.	
+ 또는 	커서를 다음 행의 처음으로 이동한다.	
H	커서를 화면의 맨 윗행으로 이동한다.	
M	커서를 화면의 중간 행으로 이동한다.	
L	커서를 화면의 맨 아랫행으로 이동한다.	
w	커서를 다음 단어의 첫 글자 위치로 이동한다.	
b	커서를 앞 단어의 첫 글자 위치로 이동한다.	
e	커서를 다음 단어의 마지막 글자 위치로 이동한다.	

## 2 – vi / vim 사용하기

- 내용 수정하기
  - [delete], [backspace]는 사용하지 않는다

명령 키	기능
r	커서가 위치한 글자를 다른 글자로 수정한다.
cw, #cw	커서 위치부터 현재 단어의 끝까지 수정한다. #에는 수정할 단어의 수를 지정한다. 예를 들 위치부터 세 단어를 수정한다.
s, #s	커서 위치부터 <b>Esc</b> 키를 입력할 때까지 수정한다. #에는 수정할 글자의 수를 지정한다. 커서 위치부터 다섯 글자를 수정한다.
cc	커서가 위치한 행의 내용을 모두 수정한다.
C	커서 위치부터 행의 끝까지 수정한다.

## 2 – vi / vim 사용하기

- 내용 삭제하기

- [입력 모드]에서는 [delete], [back space]를 사용할 수 있다
- [명령어 모드]에서는 아래와 같은 명령 키를 사용해야 한다

명령 키	기능
x, #x	커서 위치의 글자를 삭제한다. #에는 삭제할 글자 수를 지정한다. 예를 들어 3x는 세 글자를 삭제한다.
dw, #dw	커서 위치의 단어를 삭제한다. #에는 삭제할 단어 수를 지정한다.
dd, #dd	커서 위치의 행을 삭제한다. #에는 삭제할 행의 수를 지정한다. 예를 들어 5dd는 커서 위치부터 다섯 행을 삭제한다.
D( <span>Shift</span> +d)	커서 위치부터 행의 끝까지 삭제한다.

## 2 – vi / vim 사용하기

- 명령 취소하기

- [명령어 모드]에서 명령어 입력을 취소하다가 잘못 입력하여 [**ctrl+z**]입력했다면 당황해 하지 말고 **fg**를 입력하여 다시 명령어 모드로 돌아와서 정상적으로 작업을 할 수 있다
- 금방 내용을 삭제한 명령을 취소하려면 **u**를 입력하면 된다
- 내용을 입력하고 잠시 저장할 때는 **esc, :w**를 한다. 그리고 계속 입력하다가 마지막 저장한 것 이후로 작업한 것을 모두 취소할 때 **:e!**를 사용한다

명령 키	기능
u	명령을 취소한다.
U	해당 행에서 한 모든 명령을 취소한다.
:e!	마지막으로 저장한 내용 이후의 것을 버리고 새로 작업한다.



## 2 – vi / vim 사용하기

- 복사하기 또는 잘라서 붙여 넣기
  - dd로 행을 잘라내거나 yy로 복사한 후 p를 사용하여 붙여 넣기 한다

명령 키	기능
yy, #yy	커서가 위치한 행을 복사한다. #에는 복사할 행의 수를 지정한다. 예를 들어 3yy는 세 행을 복사한다.
p	커서가 위치한 행의 아래쪽에 붙인다.
P	커서가 위치한 행의 위쪽에 붙인다.
dd, #dd	커서가 위치한 행을 잘라둔다. 삭제와 같은 기능이다. #에는 잘라줄 행의 수를 지정한다. 예를 들어 3dd는 세 행을 잘라둔다.

## 2 – vi / vim 사용하기

- 입력하기: a, i, o / A, I, O
  - **a**는 해당 줄에서 커서를 우측으로 한 칸 이동하여 입력하기
  - **i**는 해당 줄 그 위치에서(이동하지 않고) 입력하기
  - **o**는 해당 줄 바로 아래에 새로운 행을 만들면서 입력하기
- **A**는 해당 줄의 제일 우측으로 커서를 이동하여 입력하기
- **I**는 해당 줄의 제일 좌측으로 커서를 이동하여 입력하기
- **O**는 해당 줄 바로 위에 새로운 행을 만들면서 입력하기



## 2 – vi / vim 사용하기

- 저장하기: w
  - **:w** → 수정중인 내용을 현재 파일에 저장
  - **:w filename** → 수정중인 내용을 새로운 파일에 저장
    - 작업 중이던 파일은 그대로 있음
    - 처음부터 vi로 시작했으면 이 파일이름으로 저장된다
- **:e!** → 수정(edit) 사항 모두 취소(!)
  - 작업하던 것을 모두 취소하는 것

## 2 – vi / vim 사용하기

- 종료하기: q
  - **:q** → 마지막 저장 후(:w) 수정 사항이 없을 경우에만 종료
  - **:wq** → 수정 사항 저장 후 종료
    - 이것과 같은 것이 바로 **ZZ**이다
  - **:q!** → 수정 사항 취소(!) 후 종료

## 2 – vi / vim 사용하기

- 검색하기

- 문자열을 검색하려면 먼저 [실행 모드(마지막 행 모드)]로 가야 한다
- 그런 다음 **/** 나 **?**을 입력하여 원하는 글자를 입력하여 검색한다
- 계속해서 다음 글자를 검색하려면 **n**을 입력한다

[표 3-15] 검색 명령 키

명령 키	기능
/문자열	문자열을 아래 방향으로 검색한다.
?문자열	문자열을 위 방향으로 검색한다.
n	원래 찾던 방향으로 다음 문자열을 찾는다.
N	역방향으로 다음 문자열을 찾는다.

- 문자 치환하기

- **:%s/검색어/치환어/g**
- cp /etc/services .
- vi services
- **:%s/tcp/TCP/g**

## 2 – vi / vim 사용하기

- 검색하기

- 문자열을 검색하려면 먼저 [실행 모드(마지막 행 모드)]로 가야 한다
- 그런 다음 **/** 나 **?**을 입력하여 원하는 글자를 입력하여 검색한다
- 계속해서 다음 글자를 검색하려면 **n**을 입력한다

명령 키	기능
/문자열	문자열을 아래 방향으로 검색한다.
?문자열	문자열을 위 방향으로 검색한다.
n	원래 찾던 방향으로 다음 문자열을 찾는다.
N	역방향으로 다음 문자열을 찾는다.

## 2 – vi / vim 사용하기

- 기타 vi에서 유용한 명령 키
  - 다른 파일 읽어 오기
    - Test.txt 파일을 편집하다가 /etc/hosts 파일을 불러와서 삽입할 수 있다
    - **:r** /etc/hosts를 하면 된다
  - 현재 작업 중인 파일의 편집을 마치고 다른 파일 편집하기
    - Test.txt 파일의 편집을 종료하고(**:w**) 곧장 sample.txt 파일을 편집하려면 **:e** sample.txt를 하면 된다
- vi에서 shell 명령어 사용하기
  - Test.txt 파일을 편집하다가 파일 목록을 확인하거나 프로그램을 컴파일을 할 수 있다
  - **:! ls -al**

명령 키	기능
! 셸 명령	vi 작업을 잠시 중단하고 셸 명령을 실행한다(vi로 돌아오려면 <b>Enter</b> 키를 입력해야 한다).
:sh	vi를 잠시 빠져나가서 셸 명령을 실행한다(vi로 돌아오려면 <b>exit</b> 명령을 입력해야 한다).

## 2 – vi / vim 사용하기

- 기타 vi에서 유용한 명령 키
  - 현재 커서가 있는 위치의 **행 번호 확인하기**
    - **Ctrl + g**
  - 파일 내용의 **마지막 행으로 커저 이동하기**
    - **Shift + g**
    - **:\$**
  - **특정한 행으로 커저 이동하기**
    - **:번호**
    - **:1**
    - **:50**
  - 마지막 명령어를 취소하기
    - **u** (소문자 u)
    - **dd**로 행을 삭제했을 때 이 작업을 취소하고자 할 때 **u**를 사용하면 유용하다
  - 마지막 명령어를 다시 실행하기
    - **.**

## 2 – vi / vim 사용하기

- 기타 vi에서 유용한 명령 키
  - 문자열 바꾸기
    - 커저가 있는 행에 한 하여 linux를 Linux로 변경하기  
**:s/linux/Linux/**
    - 특정한 범위를 정하여 centos를 CentOS로 변경하기  
**:1,15s/centos/CentOS/**
    - 파일 전체를 대상으로 Linux를 LINUX로 변경하기  
**:1,\$s/Linux/LINUX/**  
**:%s/Linux/LINUX/**
  - 커저가 있는 행과 그 아래 행을 한 줄로 합치기
    - **Shift + j**

## 2 – vi / vim 사용하기

- 기타 vi에서 유용한 명령 키
  - 행에 번호 달기
    - **:set nu**
    - **:set number**
  - 붙인 번호를 다시 없애기
    - **:set nonu**
    - **:set nonumber**
  - 특정한 사용자(adminuser) 계정으로 vi 작업할 때만 자동으로 번호 달기
    - Adminuser의 Home Directory에 .exrc 파일을 생성한다
    - **su – adminuser**
    - **vi .exrc**  
**set nu**  
**set list**
  - 모든 사용자에게 대하여 vi를 할 때 자동으로 번호 달기(환경 변수 설정)
    - **EXINIT='set nu list'**
    - **export EXINIT**



## 3 – nano 사용하기

Vi에 비해 Nano의 편리성

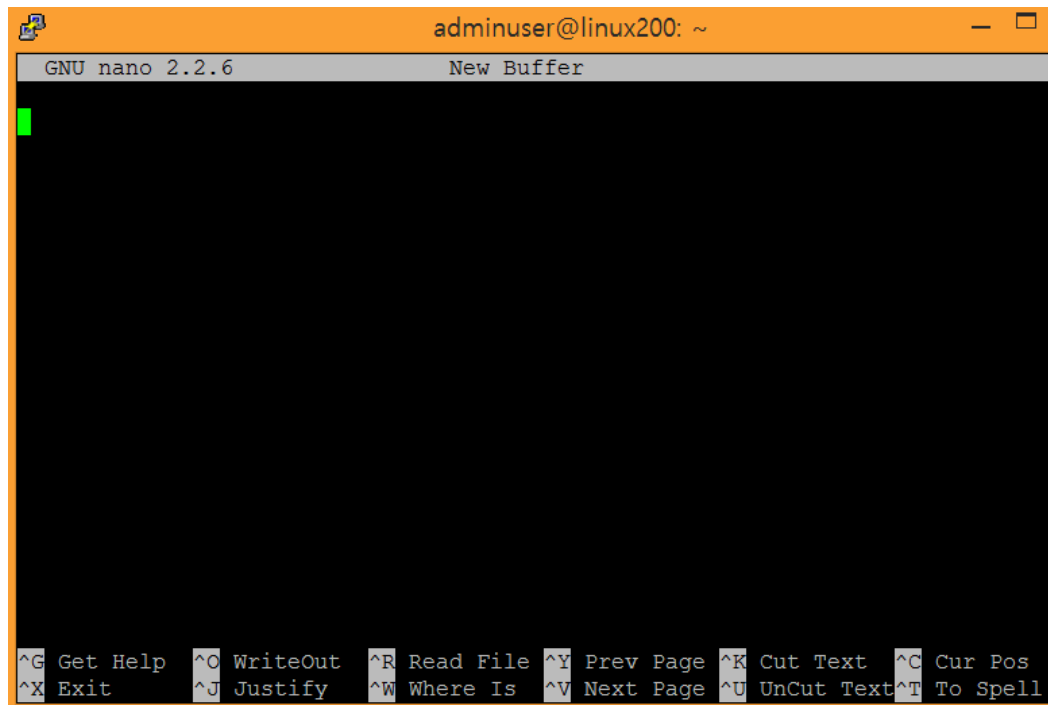
기본 명령(단축키)들

잘라내기, 복사하기, 붙여 넣기에 관련된 단축키들

화면 이동과 관련된 단축키들

# 3 – nano 사용하기

- Vi에 비해 nano의 편리성
  - Vi가 좋은 Editor이지만 처음 리눅스 사용자들에게는 어렵다
  - 편집할 때 주로 내용을 수정하기 때문에 셸 기반의 작업시 텍스트 파일을 쉽게 편집할 수 있는 것이 nano이다
- **nano**  
**nano test2.txt**



The screenshot shows a terminal window with the nano text editor running. The title bar indicates the user is 'adminuser@linux200' and the current directory is '~'. The nano header shows 'GNU nano 2.2.6' and 'New Buffer'. The editor area is mostly black with a green cursor at the top left. The bottom status bar displays various keyboard shortcuts: ^G Get Help, ^O WriteOut, ^R Read File, ^Y Prev Page, ^K Cut Text, ^C Cur Pos, ^X Exit, ^J Justify, ^W Where Is, ^V Next Page, ^U UnCut Text, and ^T To Spell.

# 3 – nano 사용하기

- 기본 명령(단축키)들

- F1을 눌러서 `ctrl+v`를 하면 다음 화면이 나오는데 여기서 단축키들을 알 수 있다

단축키	동작
<code>ctrl+g (F1)</code>	도움말 표시
<code>ctrl+x (F2)</code>	<code>nano</code> 종료 (혹은 현재의 <code>file buffer</code> 를 닫음)
<code>ctrl+o (F3)</code>	현재 편집 중인 파일 저장
<code>ctrl+j (F4)</code>	문단을 <code>justify</code> (행의 끝을 나란히 맞추다)한다. 즉, 한 문단을 한 줄로 붙인다.
<code>ctrl+r (F5)</code>	현재 <code>file</code> 에 다른 <code>file</code> 의 내용을 추가한다.
<code>ctrl+w (F6)</code>	<code>text</code> 검색
<code>ctrl+c (F11)</code>	현재의 <code>cursor</code> 위치 표시하기
<code>ctrl+t (F12)</code>	<code>spell check</code> 시작
<code>ctrl+\</code>	<code>search and replace</code>

## 3 – nano 사용하기

- 잘라내기, 복사하기, 붙여 넣기에 관련된 단축키들

단축키	동작
<b>ctrl+k (F9)</b>	현재의 <b>line</b> 혹은 선택된 <b>text</b> 삭제(그리고 저장( <b>copy</b> ))
<b>ctrl+u (F10)</b>	붙여넣기 ( <b>paste</b> )
<b>ctrl+6</b>	현재 <b>cursor</b> 위치부터 <b>text</b> 선택 시작. 이후 <b>alt+6</b> 로 복사 후 선택 종료. 아니면 다시 <b>ctrl+6</b> 를 입력하면 (복사 없이)단순 종료.
<b>alt+6</b>	선택 구간 복사. 선택 구간이 없다면 현재 <b>caret</b> 이 있는 한 줄을 복사. 이후 <b>ctrl+u</b> 로 붙여넣기 할 수 있음,

## 3 – nano 사용하기

- 화면 이동과 관련된 단축키들

단축키	동작
PageUP 또는 <code>ctrl+y</code> (F7) PageDown 또는 <code>ctrl+v</code> (F8)	이전 화면 다음 화면
<code>alt+(</code> <code>alt+)</code>	현재 문단의 시작으로 현재 문단의 끝으로
<code>alt+=</code> <code>alt+-</code>	한 줄 밑으로 스크롤 한 줄 위로 스크롤
<code>ctrl+space</code> <code>alt+space</code>	한 단어 앞으로 한 단어 뒤로 (GUI모드가 아닐 경우)
<code>alt+\</code> <code>alt+/</code>	<code>file</code> 의 첫 line으로 <code>file</code> 의 마지막 line으로
<code>alt+]</code>	현재 괄호에 match되는 괄호 찾기
<code>ctrl+-</code>	줄 번호와 열을 입력한 후 그곳으로 이동

# 4 – Gedit 사용하기

- Gedit 시작하기

- Vim 과 달리 전문적인 기능은 없고, 쉽고 간편함에 초점을 맞춘 편집기
- GNOME 데스크탑 메뉴: 프로그램 - 보조 프로그램 - 텍스트 편집기

