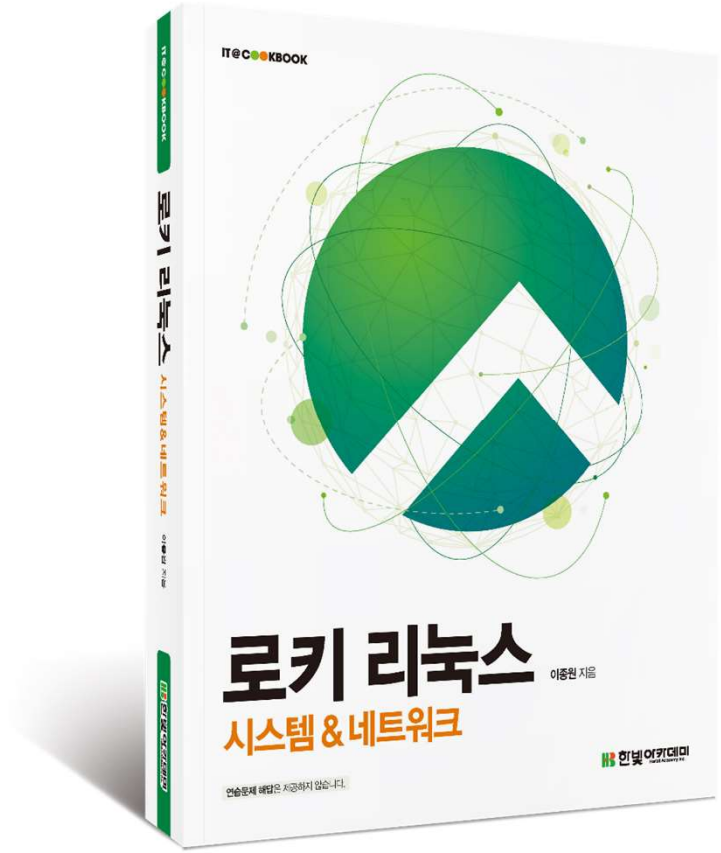


Chapter 04 문서 편집

목차

학습목표

- 리눅스에서 사용하는 편집기의 종류를 알아본다.
- 대표적인 화면 편집기인 vi를 사용할 수 있다.
- vi의 환경을 설정할 수 있다.

00 Preview

00 Preview

4장의 내용 구성

- 터미널 환경에서 사용할 수 있는 편집기의 사용법도 알아야 함
- 대표적인 문서 편집기 vim은 유닉스에서 제공하는 편집기인 vi를 업그레이드한 것
- vi의 각 모드에 대한 설명과 명령 모드의 명령키 사용 방법, 마지막 행 모드의 명령 사용 방법, vi 사용 방법, vi 환경 설정 방법으로 구성 됨

01 리눅스의 문서 편집기

01 리눅스의 문서 편집기

■ 리눅스의 편집기 종류

- 행 편집기(라인 편집기): 유닉스의 초기 편집기로 한 번에 한 행 씩만 작성하거나 수정할 수 있어 사용하기 불편
- 화면 편집기: 윈도의 메모장이나 그놈(GNOME)의 gedit처럼 화면 단위로 내용을 보고 커서를 이동하면서 작업

표 4-1 리눅스의 편집기 종류

구분	종류
행 단위 편집기	ex
화면 단위 편집기	vi(vim), nano, pico, emacs(이맥스)
GUI 편집기	gedit

01 리눅스의 문서 편집기

ex

- 행 편집기이지만 단독으로 사용하기보다는 vi에 연결하여 사용
- vi를 더욱 강력하게 하는 다양한 기능을 제공

vi(vim)

- 리눅스에서 일반적으로 사용할 수 있는 화면 편집기
- ex 편집기의 명령을 그대로 사용할 수 있고, 사용하는 명령이 매우 단순하여 빠르게 편집할 수 있어서 널리 사용
- 행 편집기에 비해 편리하지만 일반적인 윈도 메모장과는 다른 방식이기 때문에 방법을 익히기 전까지 다소 불편

01 리눅스의 문서 편집기

nano와 pico

- 워싱턴대학교에서 개발한 편집기로 초보자들이 쉽게 사용할 수 있음
- 초기 사용권이 불분명하여 GNU 프로젝트에서 pico를 기반으로 한 nano 편집기를 개발해 오픈소스로 제공. 로키 리눅스에서 nano는 제공하지만 pico는 제공하지 않음

emacs(이맥스)

- 제공하는 기능이 다양하지만 사용법이 어렵고 복잡하여 전문가 위주로 사용
- 화면 단위 이맥스 편집기 중에서는 GNU 이맥스가 가장 유명
- GNU 이맥스는 무료로 배포되지만, 보통 설치되어 있지 않아 별도 설치 필요

01 리눅스의 문서 편집기

■ 모드형과 비모드형 편집기

- 모드형은 입력 모드와 명령 모드가 구분되어 있고, 비모드형은 두 모드가 구분되어 있지 않음
- 입력 모드: 텍스트를 입력할 수 있는 모드
- 명령 모드: 텍스트를 편집(수정, 삭제, 복사, 붙이기 등)하는 모드
- 비모드형 편집기에서는 Ctrl 이나 Alt 같은 특수 키를 사용하여 편집 기능을 수행함
- 모드형 편집기는 같은 글자라도 입력 모드에서는 텍스트로 처리되어 입력되지만, 명령 모드에서는 편집 명령으로 사용

01 리눅스의 문서 편집기

■ 모드형과 비모드형 편집기

표 4-2 모드형과 비모드형 편집기의 비교

구분		모드형(vi)	비모드형(메모장)
입력 모드		텍스트 입력	
명령 모드의 예	복사	yy	Ctrl+c
	붙이기	p	Ctrl+v
	저장	:wq, ZZ	Ctrl+s
모드 전환		i, a, o, Esc	해당 없음

- 입력 모드에서는 모드형과 비모드형 모두 입력한 텍스트가 그대로 입력됨
- 명령 모드에서는 모드형의 경우 그냥 알파벳 키를 누르면 되고, 비모드형의 경우 Ctrl키와 함께 사용
- 모드형에서는 모드 전환 키가 필요하지만, 비모드형에서는 입력 모드나 명령 모드의 구분이 없으므로 모드 전환 키가 필요 없음

02 vi 사용법

02 vi 사용법

vi 동작 모드

- vi의 모드는 입력 모드와 명령 모드 외에 마지막 행 모드가 있음
- 입력 모드에서 내용을 입력하고, 명령 모드와 마지막 행 모드에서 글자와 행의 삭제·검색·저장 등의 기능을 수행함
- 마지막 행 모드에서 사용하는 명령은 ex 편집기의 명령임
- vi를 동작시킨 후 첫 입력하는 키는 모두 명령으로 해석됨
- 명령은 대문자·소문자를 구별하며, 명령 모드에서 입력 모드로 가려면 i, I, a, A, o, O 중 하나를 눌러야 함

02 vi 사용법

vi 동작 모드

- 입력 모드로 전환한 다음부터 입력하는 키는 내용으로 인식되어 버퍼에 저장됨
- 입력 모드에서 다시 명령 모드로 가려면 Esc를 눌러야 함
- 검색이나 바꾸기 같은 특별한 명령을 수행하려면 마지막 행 모드로 가야 함
- 명령 모드 상태에서 :, /, ? 중 하나를 누르면 마지막 행 모드로 전환됨
- 마지막 행 모드에서 명령을 입력하고 실행하려면 반드시 Enter를 눌러야 함
- 명령을 입력하다가 Esc를 누르면 다시 명령 모드로 돌아감
- vi의 종료는 명령 모드나 마지막 행 모드에서 할 수 있음

02 vi 사용법

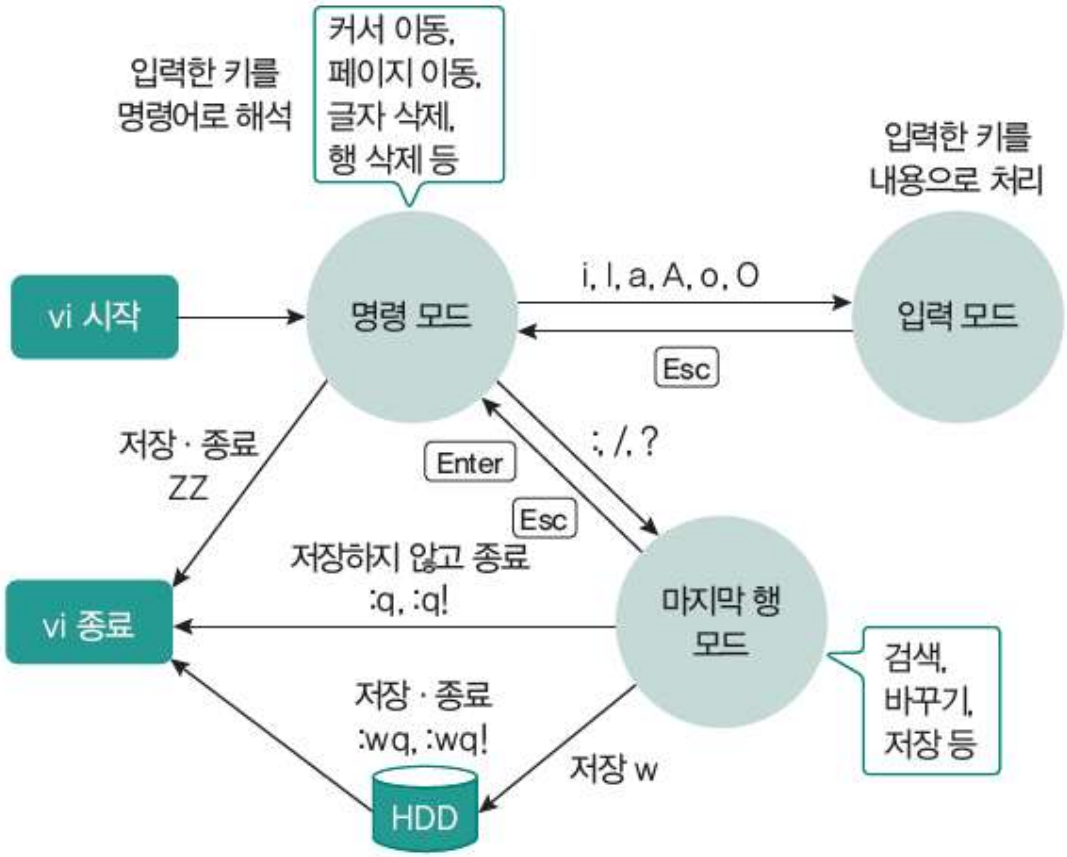


그림 4-1 vi 동작 모드

02 vi 사용법

vi 시작과 종료

vi

- 기능 지정한 파일을 편집한다. 파일명을 지정하지 않으면 빈 파일이 열리고, 이 빈 파일의 파일명은 별도로 정할 수 있다.
- 형식 vi [파일]

vi 시작하기

- vi를 시작할 때 파일을 지정하거나 지정하지 않을 수 있음
- 지정한 파일이 존재하는 파일이면 해당 파일이 열리고, 존재하지 않는 파일이면 빈 파일이 열림
- 파일을 지정하지 않으면 빈 파일이 열림
- vi를 시작할 때 파일을 지정하지 않으면 나중에 마지막 행 모드에서 파일명을 정할 수 있음

02 vi 사용법

vi 시작하기

```
[user1@localhost ~]$ vi test.txt     → test.txt라는 파일이 열린다.
                                       이 파일이 없으면 빈 파일이 열린다.

[user1@localhost ~]$ vi              → 빈 파일이 열린다. 나중에 파일명을 정할 수 있다.
```

그림 4-2 vi의 초기 화면

- vi로 파일을 열면 명령을 실행한 터미널의 전체 화면을 편집 화면으로 사용함
- 파일을 지정 하지 않고 vi를 처음 실행하면 [그림 4-2]와 같이 나타남

02 vi 사용법

vi 파일 저장하고 종료하기

- 편집을 완료하고 vi를 종료하려면 명령 모드나 마지막 행 모드에서 명령을 입력해야 함
- 작업한 내용을 저장하지 않고 종료할 수도 있고, 다른 파일명으로 저장할 수 도 있음

표 4-3 vi의 저장과 종료 명령키

모드	명령키	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 다른 파일로 저장한다.
	:wq, :wq!	작업한 내용을 저장하고 vi를 종료한다.
명령 모드	ZZ(Shift+zz)	작업한 내용을 저장하고 vi를 종료한다.

02 vi 사용법

vi 모드 전환

- vi는 처음에 명령 모드로 시작하므로 내용을 입력하려면 입력 모드로 전환해야 함

표 4-4 입력 모드 전환 명령키

명령키	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(대문자 i)	커서가 위치한 행의 첫 칼럼으로 이동하여 입력한다.
A	커서가 위치한 행의 마지막 칼럼으로 이동하여 입력한다.
O	커서가 위치한 행의 이전 행에 입력한다.

02 vi 사용법

■ i 명령키를 사용하여 입력 모드로 전환하기

- Test 디렉터리로 이동하여 ch4 폴더를 만들고 그 아래의 test.txt 파일에서 작업

```
[user1@localhost ch4]$ vi test.txt
```

- 명령 모드에서 i 명령키를 입력한 뒤 다음 내용을 입력
- 커서가 linux 다음에 위치하고, 입력 모드에서 다시 명령 모드로 전환하기 위해 Esc 를 누르면 커서가 x 위로 이동

```
Hanbit linux study          → Enter를 누르면 다음 행으로 이동한다.
I like linux                → Esc를 누르면 명령 모드로 전환된다.
~
```

```
Hanbit linux study
I like linux                → 명령 모드로 전환되고 커서가 x 위로 이동한다.
~
```

02 vi 사용법

■ o 명령키를 사용하여 입력 모드로 전환하기

- 명령 모드에서 현재 커서가 위치한 행의 다음 행에 글자를 입력하려면 o 키를 누르면 커서의 위치가 아래 행으로 이동하고 입력 모드 상태가 됨
- Esc 를 누르면 다시 명령 모드로 전환되고 커서는 아래 행의 첫 칼럼 자리에 놓임

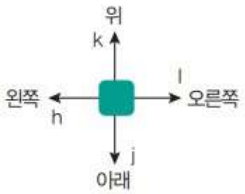
```
Hanbit linux study
I like linu Hanbit linux        ➞ 명령 모드에서 o를 입력하면 커서가 아래 행으로 이동한다.
~
```

02 vi 사용법

■ 커서 이동하기

- vi에서는 마우스로 커서를 옮길 수 없고 모두 키보드로 해야 함

표 4-5 커서 이동 명령키

명령키	기능	
k/j	커서를 한 행 위/아래로 이동한다.	
l/h	커서를 한 글자 오른쪽/왼쪽으로 이동한다.	
^ 또는 0/$	커서를 현재 행의 처음/마지막으로 이동한다.	
−/+ 또는 Enter	커서를 앞/다음 행의 처음으로 이동한다.	
H/M/L	커서를 화면의 맨 위/중간/맨 아래 행으로 이동한다.	
w	커서를 다음 단어의 첫 글자로 이동한다.	
e	커서를 현재 단어의 마지막 글자로 이동한다.	
b	커서를 앞 단어의 첫 글자로 이동한다.	

02 vi 사용법

■ 커서 이동하기

① k 명령키를 입력하면 커서가 w 바로 위에 있는 t로 이동

② j 명령키를 입력하면 커서가 w 바로 아래에 있는 공백으로 이동

③ w 명령키를 입력하면 커서가 다음 단어 be의 첫 글자인 b로 이동

④ b 명령키를 입력하면 커서가 앞 단어 you의 첫 글자인 y로 이동

⑤ e 명령키를 입력하면 커서가 현재 단어 will의 마지막 글자인 l로 이동. 만약 다시 e 명령키를 입력하면 커서가 다음 단어 be의 마지막 글자인 e로 이동

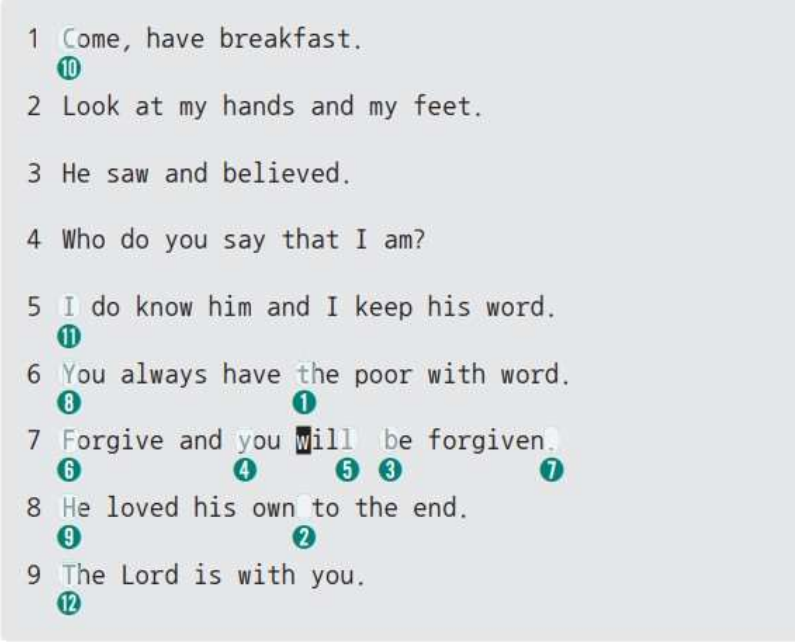


그림 4-3 커서 이동 명령키의 예

02 vi 사용법

■ 커서 이동하기

⑥ ^(캐럿) 명령키(Shift +6)를 입력하면 커서가 현재 행의 첫 글자인 F로 이동

⑦ $ 명령키(Shift +4)를 입력하면 커서가 현재 행의 끝인 .(마침표)로 이동

⑧ - 명령키를 사용하면 커서가 위 행의 첫 글자인 Y로 이동

⑨ + 명령키를 사용하면 커서가 아래 행의 첫 글자인 H로 이동

⑩ H 명령키를 입력하면 커서가 화면의 첫 행으로 이동

⑪ M 명령키를 입력하면 커서가 화면의 중간 행으로 이동

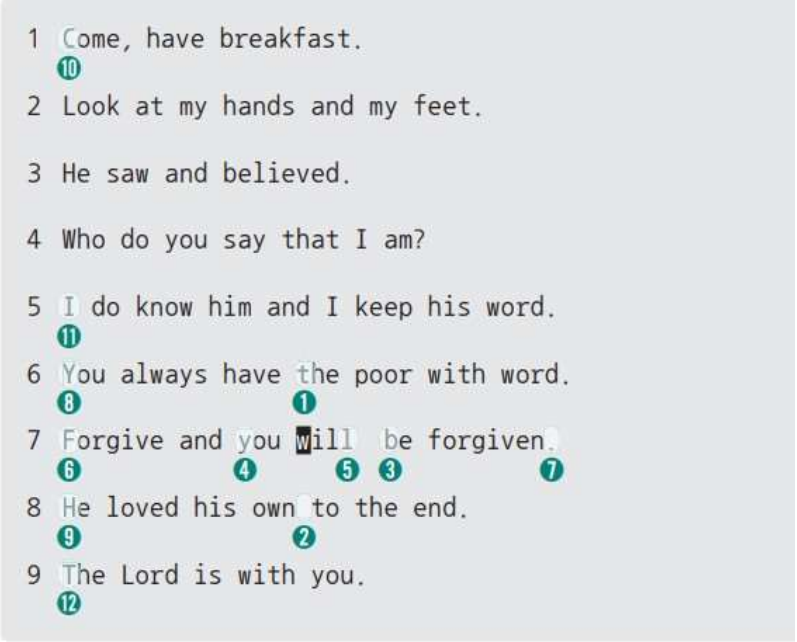


그림 4-3 커서 이동 명령키의 예

02 vi 사용법

■ 커서 이동하기

⑫ L 명령키를 입력하면 커서가 화면의 마지막 행으로 이동. 내용이 화면 크기보다 작으면 내용의 마지막 행으로 커서가 이동

⑬ h나 l(소문자 L) 명령키를 입력하면 커서가 좌우로 한 칸씩 움직임

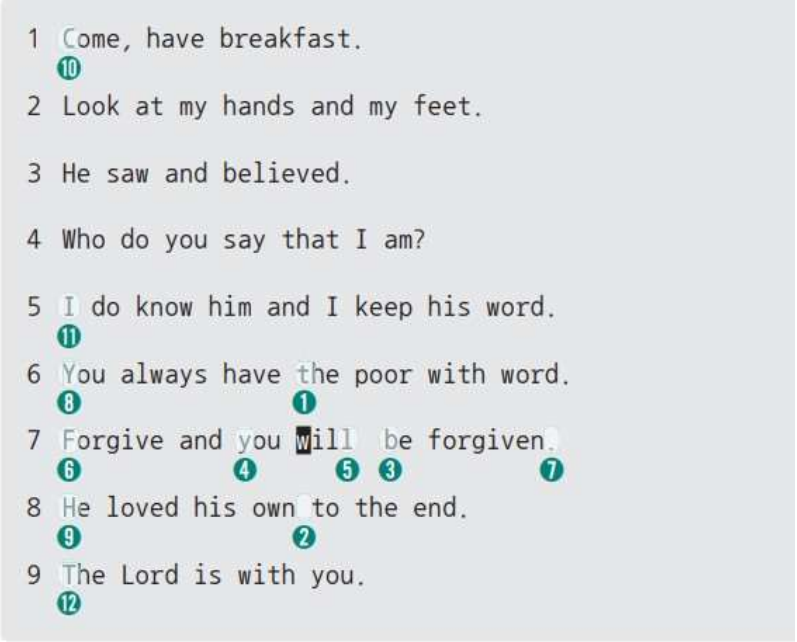


그림 4-3 커서 이동 명령키의 예

02 vi 사용법

■ 커서 이동하기

- 초기의 유닉스 vi는 방향키로 커서를 이동할 수 없었지만 리눅스 vi는 방향키로도 커서를 이동할 수 있음

표 4-6 기존 vi 명령키와 방향키, Home · End

명령키	방향키	명령키	방향키, Home · End
k	위 방향키(↑)	h	왼쪽 방향키(←)
j	아래 방향키(↓)	^ 또는 0	Home
l	오른쪽 방향키(→)	$	End

02 vi 사용법

■ 화면 이동하기

- vi로 한 번에 볼 수 있는 내용은 터미널의 화면 크기에 따라 달라짐
- 파일 크기가 터미널의 화면 크기보다 크면 화면을 이동해야 함

표 4-7 화면 이동 명령키

기존 명령키	기능	추가 명령키
^u(Ctrl+u)	반 화면 위로 이동한다.	
^d(Ctrl+d)	반 화면 아래로 이동한다.	
^b(Ctrl+b)	한 화면 위로 이동한다.	Page Up
^f(Ctrl+f)	한 화면 아래로 이동한다.	Page Down
^y(Ctrl+y)	화면을 한 행만 위로 이동한다.	
^e(Ctrl+e)	화면을 한 행만 아래로 이동한다.	

02 vi 사용법

■ 특정 행으로 바로 이동하기

표 4-8 특정 행으로 바로 이동하는 명령키

명령키	기능
G(Shift+g)	파일의 마지막 행으로 커서가 이동한다.
행 번호G(Shift+g)	지정한 행 번호로 커서가 이동한다.
:행 번호	지정한 행 번호로 커서가 이동한다(마지막 행 모드).
:$	파일의 마지막 행으로 커서가 이동한다(마지막 행 모드).

- 명령 모드에서 G 명령키만 입력하면 파일의 마지막 행으로 이동하고, G 명령키 앞에 행 번호를 붙이면 해당 행으로 이동
- 마지막 행 모드 명령을 사용하려면 :을 입력하여 마지막 행으로 전환한 뒤 이동하려는 행 번호를 입력하고 Enter 를 누르면 됨
- 행 번호 대신 $를 입력하면 파일의 마지막 행으로 이동

02 vi 사용법

■ 내용 수정

- 입력 된 내용을 수정하는 명령키에는 한 글자만 수정하거나, 단어별로 수정하거나, 수정할 글자 수를 지정해서 수정하는 명령키가 있음
- 한 글자를 수정하는 r 명령키 외의 다른 명령키는 자동으로 입력 모드로 전환되기 때문에 수정을 마치면 Esc 를 눌러서 명령 모드로 돌아가야 수정이 완료

표 4-9 내용 수정 명령키

명령키	기능
r	커서가 위치한 글자를 다른 글자로 수정한다.
cw, #cw	커서 위치부터 현재 단어의 끝까지 수정한다. #에는 수정할 단어의 수를 지정한다. 예를 들어 3cw는 커서 위치부터 세 단어를 수정한다.
s, #s	커서 위치부터 Esc 를 입력할 때까지 수정한다. #에는 수정할 글자의 수를 지정한다. 예를 들어 5s는 커서 위치부터 다섯 글자를 수정한다.
cc	커서가 위치한 행의 내용을 모두 수정한다.
C	커서 위치부터 행의 끝까지 수정한다.

02 vi 사용법

■ 한 글자 수정하기: r 명령키

- 수정하려는 글자 위에 커서를 놓은 후 r 명령키를 먼저 입력하고 바꾸려는 새 글자를 입력하면 됨

```
Hanbit winux study   → r 명령키로 글자를 수정한다(l→w).
I like linu Hanbit linux


~
```

02 vi 사용법

■ 단어 수정하기: cw, #s 명령키

- 한 단어를 바꿀 때는 cw 명령키나 #s 명령 키를 사용
- vi에서는 단어를 공백문자나 특수문자로 구분

```
Hanbit study          → cw 명령키 입력 시 winux가 사라진다.
I like linu Hanbit linux


~
(생략)
~
— 끼워넣기 —
```

- 띄어쓰기 없이 붙어 있는 winux를 한 단어로 인식
- 한 단어이므로 cw 명령키를 입력하거나, 글자 수가 다섯 개이므로 5s로 수정
- w 위치에서 cw나 5s를 먼저 입력하면 수정할 대상이 지워지고 커서가 입력을 기다림

02 vi 사용법

■ 단어 수정하기: cw, #s 명령키

- 바꿀 단어의 글자 수는 기존 단어의 글자 수와 상관없이 지정할 수 있음
- cw나 #s 명령키를 사용할 때는 반드시 마지막에 Esc 를 눌러야 명령 모드로 전환되어 수정이 완료됨
- 수정이 끝나면 커서는 마지막 글자인 r 위에 놓임

```
Hanbit editor study        → 수정 완료 후 Esc를 눌러야 명령 모드로 전환된다.
I like linu Hanbit linux
```

02 vi 사용법

■ 행 단위 수정하기: C 명령키

- 현재 커서가 위치한 r부터 행의 끝까지 수정하려면 C 명령키를 사용
- r부터 행의 마지막까지 모두 지워지고 입력 모드로 전환되어 입력을 기다림

```
Hanbit edito          → r부터 모두 지워지고 입력 모드로 전환된다.
I like linu Hanbit linux
```

- 수정할 대상의 글자 수와 상관없이 원하는 대로 입력하면 됨
- r이 없어졌으므로 'r vi'로 수정하고 Esc 를 눌러 명령 모드로 전환

```
Hanbit editor vi
I like linu Hanbit linux
```

02 vi 사용법

■ 행 단위 수정하기: cc 명령키

- cc 명령키를 입력하면 현재 행의 모든 내용이 삭제되고, 커서가 행의 처음으로 이동하여 새로운 입력을 기다림

```
                    → 모두 지워지고 행의 처음으로 이동하여 입력 모드로 전환된다.
I like linu Hanbit linux
```

- 원하는 내용을 입력하고 Esc 를 눌러 명령 모드로 전환하면 수정이 완료

```
Hanbit editor vi study
I like linu Hanbit linux
```

02 vi 사용법

■ 내용 삭제

- 입력 모드에서 내용을 입력하는 도중 틀린 글자를 삭제할 때는 Back Space 나 Delete 를 사용하면 됨
- 명령 모드에서 글자나 행을 삭제하려면 명령키를 사용해야 함
- 글자나 행의 수를 숫자로 지정

표 4-10 내용 삭제 명령키

명령키	기능
x, #x	커서 위치의 글자를 삭제한다. #에는 삭제할 글자 수를 지정한다.
dw, #dw	커서 위치의 단어를 삭제한다. #에는 삭제할 단어 수를 지정한다.
dd, #dd	커서 위치의 행을 삭제한다. #에는 삭제할 행의 수를 지정한다.
D(Shift+d)	커서 위치부터 행의 끝까지 삭제한다.

02 vi 사용법

■ 내용 삭제

- 현재 커서가 놓인 l 한 글자만 지우려면 x 명령키를 입력

```
Hanbit editor vi study
I like linu Hanbit linux
```

```
Hanbit editor vi study
I like inu Hanbit linux     → x 명령키로 커서 위치의 l을 삭제한다.
```

- 나머지 inu를 모두 지우려면 글자 수를 지정하여 3x를 입력하거나 단어를 지우는 dw 명령키를 사용하면 됨

```
Hanbit editor vi study
I like Hanbit linux         → dw로 삭제하면 커서가 다음 단어의 첫 글자로 이동한다.
```

```
Hanbit editor vi study
I like Hanbit linux         → 3x로 삭제하면 커서가 공백문자에 위치한다.
```

02 vi 사용법

내용 삭제

- 커서가 위치한 현재 행을 지우려면 dd 명령키를 입력

```
Hanbit editor vi study
```

02 vi 사용법

■ 명령 취소

- 삭제나 수정을 잘못한 경우 이전 명령을 취소할 수 있음
- vi에서 Ctrl +z는 취소가 아니라 vi를 잠시 중단하고 셸로 돌아가는 명령

표 4-11 이전 명령 취소 명령키

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

- 이전 명령을 취소하려면 u 명령키. vi에는 히스토리 기능이 있어서 실행했던 역순으로 취소할 수 있음

```
Hanbit editor vi study
I like linu Hanbit linux
```

- 모든 작업을 취소하고 원래대로 복구하려면 U 명령키를 이용
- :e!는 마지막으로 저장한 내용 이후의 작업 내용을 모두 무시하고 재편집 할 때 사용

02 vi 사용법

■ vi로 입력·수정·삭제·복구하기

① vi로 새로운 파일인 exec.txt 파일을 연다

```
[user1@localhost ch4]$ vi exec.txt
```

② i 키를 눌러 입력 모드로 전환하고 다음 내용을 입력

```
Good morning everyone.
Nice to meet you.
I am a linux beginner.
Now introduce yourself.
```

③ Esc 를 입력하여 명령 모드로 전환하고 :w 명령으로 파일 내용을 저장함

```
Good morning everyone.
Nice to meet you.
I am a linux beginner.
Now introduce yourself.
~
:w
```

02 vi 사용법

■ vi로 입력·수정·삭제·복구하기

④ 커서를 3행의 beginner로 이동. 3G와 l 명령키(또는 → 나 w 명령키와 h 명령키)를 사용하여 이동할 수 있음

```
Good morning everyone.
Nice to meet you.
I am a linux beginner.
Now introduce yourself.
```

⑤ beginner를 expert로 수정. cw나 8s 명령키를 사용. 수정한 후에는 Esc를 눌러 명령 모드로 전환

```
Good morning everyone.
Nice to meet you.
I am a linux expert.
Now introduce yourself.
```

02 vi 사용법

■ vi로 입력·수정·삭제·복구하기

⑥ - 명령키로 2행의 첫 글자인 N으로 이동

```
Good morning everyone.
Nice to meet you.
I am a linux expert.
Now introduce yourself.
```

⑦ w 명령키로 커서를 meet로 이동하고, 단어 meet를 dw 명령키로 삭제

```
Good morning everyone.
Nice to you.
I am a linux expert.
Now introduce yourself.
```

02 vi 사용법

■ vi로 입력·수정·삭제·복구하기

⑧ 단어 y부터 행의 끝까지 D(Shift +d) 명령키로 삭제

```
Good morning everyone.
Nice to
I am a linux expert.
Now introduce yourself.
```

⑨ U 명령키로 2행에서 한 모든 삭제를 취소

```
Good morning everyone.
Nice to meet you.
I am a linux expert.
Now introduce yourself.
```

02 vi 사용법

■ vi로 입력·수정·삭제·복구하기

⑩ :wq 명령으로 파일 내용을 저장하고 종료

```
Good morning everyone.
Nice to meet you.
I am a linux expert.
Now introduce yourself.
~
:wq
```

혼자해보기 vi로 입력 · 수정 · 삭제 · 저장하기

❶ exec.txt 파일을 연다.
❷ 현재 행의 아래 행에 다음 내용을 추가한다.
Welcome Linux World.
❸ 커서를 4행의 첫 글자인 I로 이동한다.
❹ I am을 We are로 수정한다.
❺ 커서를 마지막 행으로 이동한다.
❻ 마지막 행을 삭제한다.
❼ 명령 모드의 명령을 사용하여 파일을 저장하고 종료한다.

02 vi 사용법

■ 복사하기, 잘라내기, 붙이기

- 복사하기, 잘라내기, 붙이기를 할 때 마우스가 아닌 명령키를 사용해야 함

표 4-12 복사하기, 잘라내기, 붙이기 명령키

명령키	기능
yy, #yy	커서가 위치한 행을 복사한다. #에는 복사할 행의 수를 지정한다.
p	커서가 위치한 행의 아래쪽에 붙인다.
P(대문자 P)	커서가 위치한 행의 위쪽에 붙인다.
dd, #dd	커서가 위치한 행을 잘라둔다. 삭제와 같은 기능이다. #에는 잘라둘 행의 수를 지정한다.

- 복사하기나 잘라내기를 하면 해당 내용이 임시 버퍼에 저장됨
- 복사하기나 잘라내기를 한 뒤에는 다른 명령을 사용하지 말고 즉시 원하는 위치로 이동하여 붙이기를 하는 것이 좋음. 다른 명령을 사용하다 임시 버퍼에 저장된 내용을 잃어버릴 수도 있기 때문

02 vi 사용법

■ 복사하기, 잘라내기, 붙이기

- 빈 행을 dd 명령으로 삭제하고 파일의 처음 위치로 이동

```
Hanbit editor vi study
I like linu Hanbit linux
```

- 현재 커서가 있는 행만 복사하려면 그냥 yy 명령키만 입력. 1행과 2행을 함께 복사 하려면 2yy를 입력

```
Hanbit editor vi study
I like linu Hanbit linux
Hanbit editor vi study
I like linu Hanbit linux
```

- dd 명령키는 삭제 뿐 아니라 잘라내기를 할 때도 사용

```
Hanbit editor vi study
Hanbit editor vi study
I like linu Hanbit linux
I like linu Hanbit linux
```

02 vi 사용법

■ 네임드 버퍼 사용하기

- yy 명령키로 복사하거나 dd 명령키로 잘라내기를 할 경우, 복사하거나 잘라낸 내용이 버퍼에 저장됨
- 버퍼는 복사하거나 잘라낸 내용을 임시로 저장해 두는 공간으로 윈도의 클립보드와 같은 기능을 수행
  - 언네임드 버퍼: 이름을 붙이지 않은 버퍼. 내용을 하나만 저장할 수 있음
  - 네임드 버퍼: 이름을 붙여서 사용할 수 있는 버퍼. 각각 다른 이름을 붙인 버퍼에 독립적으로 내용을 저장하고 사용할 수 있음
- 네임드 버퍼에 이름을 붙일 때는 "+문자 또는 "+숫자의 형태로 사용. 숫자를 사용할 경우 숫자 버퍼라고 함
  - 네임드 버퍼: "a, "b, "c, "d, ..., "z
  - 숫자 버퍼: "1, "2, ..., "9

02 vi 사용법

■ 네임드 버퍼 사용하기

- 네임드 버퍼에 커서가 위치한 행을 저장하려면 '버퍼 이름+yy'를 입력
- 커서가 위치한 행의 아래쪽에 붙이려면 "ap와 같이 '버퍼 이름+p'를 입력

```
Hanbit editor vi study
Hanbit editor vi study
I like linu Hanbit linux
I like linu Hanbit linux
```

- 2행을 "a 버퍼에 잘라서 저장

```
Hanbit editor vi study
I like linu Hanbit linux
I like linu Hanbit linux
```

02 vi 사용법

■ 네임드 버퍼 사용하기

- 2행을 잘라 "b 버퍼에 저장

① 커서를 1행으로 이동

② "b 버퍼의 내용을 1행 위 행에 붙여 넣으려면 "bP를 입력

③ "a 버퍼의 내용을 1행 아래 행에 붙여 넣으려면 "ap를 입력

```
I like linu Hanbit linux
Hanbit editor vi study
Hanbit editor vi study
I like linu Hanbit linux
```

02 vi 사용법

■ 범위 지정하기

- 범위 지정은 마지막 행 모드에서 할 수 있으며 특수 문자(., $, %)와 숫자를 사용
- .(마침표)는 커서가 위치한 현재 행을, $는 마지막 행을, %는 전체 행을 의미. 숫자로 행 번호나 행의 개수를 지정할 수 있음
- 범위를 지정 하려면 :(콜론)을 입력하여 마지막 행 모드로 전환하고 명령키를 사용
- 명령키에서 여러 행을 지정하는 경우, 숫자와 숫자 사이의 ,(쉼표)는 띄어쓰기를 하지 않음

표 4-13 범위 지정 명령키

명령키	기능
1,$ 또는 %	1행부터 마지막 행까지 지정한다.
1,.	1행부터 커서가 있는 행까지 지정한다.
.,$	커서가 있는 행부터 마지막 행까지 지정한다.
.,-3	현재 행과 이전 세 행까지(총 네 행) 지정한다.
10,20	10행부터 20행까지 지정한다.
10	10행만 지정한다.

02 vi 사용법

■ 마지막 행 모드에서 복사하기, 잘라내기, 붙이기

- 잘라내기 명령은 삭제 명령과 동일
- 명령키 앞에 붙은 :은 마지막 행 모드로 이동하기 위한 것으로, 이미 마지막 행 모드 상태라면 :을 제외하고 명령키를 입력

표 4-14 마지막 행 모드에서의 복사하기, 잘라내기, 붙이기 명령키

명령키	기능
:#y	#로 지정한 행을 복사한다. 예를 들면 3y는 세 번째 행을 복사한다.
:〈범위〉y	범위로 지정한 행을 복사한다. 예를 들면 2,4y는 2~4행을 복사한다.
:#d	#로 지정한 행을 잘라낸다(삭제). 예를 들면 3d는 세 번째 행을 잘라낸다.
:〈범위〉d	범위로 지정한 행을 잘라낸다(삭제). 예를 들면 1,4d는 1~4행을 잘라낸다.
:pu	현재 행 다음에 버퍼의 내용을 붙인다.
:#pu	#로 지정한 행 다음에 버퍼의 내용을 붙인다. 예를 들면 4pu와 같이 지정한다.

02 vi 사용법

■ 마지막 행 모드에서 복사하기, 잘라내기, 붙이기

- 범위 지정과 함께 마지막 행 모드에서 사용하는 명령을 입력하여 살펴보기

```
I like linu Hanbit linux
Hanbit editor vi study
Hanbit editor vi study
I like linu Hanbit linux
```

- :2,3d 명령키로 2행과 3행을 잘라내고 :2pu 명령키로 2행 다음 잘라낸 것을 붙이면 다음과 같이 바뀜

```
I like linu Hanbit linux
I like linu Hanbit linux
Hanbit editor vi study
Hanbit editor vi study
```

02 vi 사용법

■ 마지막 행 모드에서 복사하기, 잘라내기, 붙이기

- :1,2y 명령키로 1행부터 2행까지 복사하고 :4pu 명령키로 붙여넣기

```
I like linu Hanbit linux
I like linu Hanbit linux
Hanbit editor vi study
Hanbit editor vi study
I like linu Hanbit linux
I like linu Hanbit linux
```

02 vi 사용법

■ 검색하기

- 문자열을 검색하려면 마지막 행 모드로 가야 함
- 검색하기 위해 마지막 행으로 이동할 때는 :이 아니라 /나 ?를 입력
- /는 커서의 위치에서 아래 방향으로 검색하고, ?는 커서의 위치에서 위 방향으로 검색
- /나 ? 다음에 찾고자 하는 문자열을 입력하고 Enter 를 누르면 검색이 진행되고, 해당 문자열을 찾으면 그 문자열의 시작 위치로 커서가 이동
- 계속 다음 문자열을 검색하려면 nnext 명령을 사용

표 4-15 검색 명령키

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

02 vi 사용법

■ 검색하기

- 커서가 6행에 있는 상태에서 문자열을 검색하기 위해 /를 입력하면 마지막 행 모드로 전환
- 검색할 문자열인 'Hanbit'을 입력하고 Enter를 누르면 커서 위치보다 뒤 쪽에 위치한 같은 행의 Hanbit으로 커서가 이동

```
I like linu Hanbit linux
I like linu Hanbit linux
Hanbit editor vi study
Hanbit editor vi study
I like linu Hanbit linux
I like linu Hanbit linux
~
(생략)
/Hanbit
```

02 vi 사용법

■ 검색하기

- 계속 Hanbit을 검색하려고 n을 입력하면 6행이 파일의 마지막 행이므로 '끝까지 찾았음, 처음부터 계속'이라는 메시지를 출력하고 1행의 Hanbit으로 커서가 이동

```
I like linu Hanbit linux
I like linu Hanbit linux
Hanbit editor vi study
Hanbit editor vi study
I like linu Hanbit linux
I like linu Hanbit linux
~
(생략)
끝까지 찾았음, 처음부터 계속
```

02 vi 사용법

■ 바꾸기

- 기존 문자열을 다른 문자열로 바꾸려면 먼저 :을 입력하여 마지막 행 모드로 전환
- 문자열을 바꾸는 명령을 사용하여 커서 위치의 문자열만 바꿀 수도 있고, 파일 전체나 특정 범위 내에서 해당하는 문자열을 모두 찾아 바꿀 수도 있음

표 4-16 바꾸기 명령키

명령키	기능
:s/문자열1/문자열2/	커서가 위치한 행에서 첫 번째로 나오는 문자열1을 문자열2로 바꾼다.
:%s/문자열1/문자열2/g	파일 전체에서 모든 문자열1을 문자열2로 바꾼다.
:〈범위〉s/문자열1/문자열2/	범위 내 모든 각 행에서 첫 번째로 나오는 문자열1을 찾아 문자열2로 바꾼다.
:〈범위〉s/문자열1/문자열2/g	범위 내 모든 행에서 문자열1을 문자열2로 바꾼다.
:〈범위〉s/문자열1/문자열2/gc	범위 내 모든 행에서 문자열1을 문자열2로 바꿀 때 수정할지 여부를 묻는다.

02 vi 사용법

■ 바꾸기

- 커서가 위치한 행의 Hanbit을 HANBIT으로 바꾸는 예

```
I like linu HANBIT linux
I like linu Hanbit linux
Hanbit editor vi study
Hanbit editor vi study
I like linu Hanbit linux
I like linu Hanbit linux
~
(생략)
:s/Hanbit/HANBIT/
```

02 vi 사용법

■ 바꾸기

- 범위를 정해 문자열을 바꾸는 예

```
I like linu HANBIT linux
I like linu Hanbit linux
Hanbit Hanbit vi study
Hanbit Hanbit vi study
I like linu Hanbit linux
I like linu Hanbit linux
~
(생략)
:3,4s/editor/Hanbit/
```

- 3~4행에 있는 editor를 Hanbit으로 바꾸는 명령은 :3,4s/editor/Hanbit/

02 vi 사용법

■ 바꾸기

- 명령키에서 마지막에 g가 있을 때와 없을 때의 차이를 살펴보기

```
I like linu HANBIT linux
I like linu Hanbit linux
HANBIT Hanbit vi study
HANBIT HANBIT vi study
I like linu Hanbit linux
I like linu Hanbit linux
~
(생략)
:s/Hanbit/HANBIT/g
```

- 3행에서 :s/Hanbit/HANBIT/를 수행하고 4행에서는 :s/Hanbit/HANBIT/g를 수행한 결과
- g가 있으면 해당 행이나 범위 내에서 바꾸려는 문자열을 모두 찾아 다른 문자열로 바꿈

02 vi 사용법

■ 바꾸기

- 파일 전체에서 문자열을 바꿔보기

```
I like linu HANBIT linux
I like linu HANBIT linux
HANBIT HANBIT vi study
HANBIT HANBIT vi study
I like linu HANBIT linux
I like linu HANBIT linux
~
(생략)
4 substitutions on 4 lines
```

- Hanbit을 HANBIT으로 바꿀 때는 :%s/Hanbit/HANBIT/g 또는 :1,$s/Hanbit/HANBIT/g를 사용하면 됨
- 명령을 실행 하면 몇 개가 바뀌었는지 마지막 행에 출력됨

02 vi 사용법

■ vi 편집 방법 익히기

① vi로 새로운 파일인 exec2.txt 파일을 연다

```
[user1@localhost ch4]$ vi exec2.txt
```

② i 키를 눌러 입력 모드로 전환하고 다음 내용을 입력

```
Good morning everyone.
My name is Gildong Hong.
This is a living room.
```

③ Esc 를 입력하여 명령 모드로 전환하고 :w 명령으로 파일 내용을 저장

④ 1행을 복사하여 3행 다음에 붙임

```
Good morning everyone.
My name is Gildong Hong.
This is a living room.
Good morning everyone.    ➜ ① 1G 입력 → ② yy 입력 → ③ jj 입력 → ④ p 입력
```

02 vi 사용법

■ vi 편집 방법 익히기

⑤ 파일에서 morning이란 단어를 검색

```
Good morning everyone.
My name is Gildong Hong.
This is a living room.
Good morning everyone.          ➞ ① /morning Enter입력 → ② n 입력
~
/morning
```

⑥ 파일에서 morning이란 단어를 모두 찾아 afternoon으로 바꿈

```
Good afternoon everyone.
My name is Gildong Hong.
This is a living room.
Good afternoon everyone.        ➞ :%s/morning/afternoon/g 입력
~
:%s/morning/afternoon/g
```

⑦ 저장한 후 종료

02 vi 사용법

■ vi 편집 방법 익히기

혼자해보기 vi 편집 방법 익히기

❶ vi로 exec2.txt 파일을 연다.

❷ "a 버퍼에 1행을, "b 버퍼에 2행을, "c 버퍼에 3행을 저장한다.

❸ 4행을 삭제한다.

❹ 네임드 버퍼에 저장한 내용을 사용하여 파일을 다음과 같이 수정한다.

```
Good afternoon everyone.
This is a living room.
My name is Gildong Hong.
This is a living room.
My name is Gildong Hong.
Good afternoon everyone.
```

❺ 첫 번째로 나오는 Hong을 찾아 Park으로 바꾼다.

❻ 파일을 저장하고 종료한다.

02 vi 사용법

■ 파일 읽어오기, 여러 파일 편집하기

- 현재 작업 중인 파일에 다른 파일을 읽어 들이거나, 파일 작업을 마친 뒤 vi를 종료하지 않고 다른 파일로 작업을 전환할 수 있음
- vi를 시작할 때 여러 개의 파일명을 지정하고 차례로 다음 파일로 이동하면서 작업할 수도 있음

표 4-17 파일 관련 명령키

명령키	기능
:r 파일	지정한 파일을 읽어들여 현재 커서의 다음 행에 삽입한다.
:e 파일	지정한 파일로 전환한다(기존 파일을 :w로 저장한 뒤에 실행해야 한다).
:n 파일	vi 시작 시 여러 파일을 지정했을 경우 다음 파일로 작업을 이동한다.

02 vi 사용법

■ 다른 파일 읽어오기

- 파일을 편집하는 도중에 다른 파일을 읽어 들이는 경우

```
I like linu HANBIT linux
I like linu HANBIT linux
HANBIT HANBIT vi study
HANBIT HANBIT vi study
I like linu HANBIT linux
I like linu HANBIT linux
```

- exec2.txt 파일을 읽어 들이기 위해서 :r exec2.txt를 실행하면 exec2.txt 파일의 내용이 test.txt 파일의 4행 다음에 삽입 됨

02 vi 사용법

■ 다른 파일 읽어오기

```
I like linu HANBIT linux
I like linu HANBIT linux
HANBIT HANBIT vi study
HANBIT HANBIT vi study
Good afternoon everyone.
This is a living room.
My name is Gildong Park.
This is a living room.
My name is Gildong Hong.
This is a living room.

I like linu HANBIT linux
I like linu HANBIT linux
```

02 vi 사용법

■ 파일 편집을 마치고 다른 파일 편집하기

- :e 명령키는 현재 작업 중인 파일의 작업을 마치고 다른 파일을 편집하려고 할 때 사용함

```
I like linu HANBIT linux
I like linu HANBIT linux
HANBIT HANBIT vi study
HANBIT HANBIT vi study
Good afternoon everyone.
(생략)
~
E37: 마지막으로 고친 뒤 저장되지 않았습니다 (무시하려면 ! 더하기)
```

- test.txt 파일 편집을 완료하고 exec.txt 파일 편집으로 바꾸려면 :e exec. txt를 입력하면 됨
- 작업 중인 파일을 저장하지 않고 :e exec.txt를 실행하면 다음과 같은 오류 메시지가 출력
- 이 경우 파일의 작업 내용을 저장하고 다른 파일로 이동하려면 :w로 먼저 저장하고 :e 명령을 사용
- 작업한 내용을 저장하지 않고 다른 파일로 이동하려면 :e! exec.txt를 입력

02 vi 사용법

■ 여러 파일 편집하기

- vi를 시작할 때 다음과 같이 파일명을 여러 개 지정할 수 있음

```
[user1@localhost ch4]$ vi test.txt exec.txt exec2.txt
```

- 앞의 예는 파일 세 개를 모두 작업하겠다는 뜻
- 맨 앞에 지정한 파일이 먼저 열리고, 이 파일의 작업을 마친 후 다음 파일로 이동하려면 :n을 입력
- 파일을 수정했으면 저장한 후 :n을 입력
- :n 명령키로 다음 파일로 이동한 다음에는 다시 이전 파일로 돌아갈 수 없으므로 vi를 종료하고 다시 시작하거나 :e 명령키를 사용해야 함

02 vi 사용법

■ vi에서 셸 명령 사용하기

- vi로 파일을 편집하던 도중 작업 디렉터리의 파일 목록을 확인하거나 프로그램을 컴파일하는 등 셸 명령을 실행해야 하는 경우가 있음
- vi를 종료하지 않고 셸 명령을 실행하려면 다음 명령키를 사용

표 4-18 셸 명령 실행 명령키

명령키	기능
:! 셸 명령	vi 작업을 잠시 중단하고 셸 명령을 실행한다(vi로 돌아오려면 Esc를 누른다).
:sh	vi를 잠시 빠져나가서 셸 명령을 실행한다(vi로 돌아오려면 exit 명령을 입력한다).

02 vi 사용법

:! 명령키 이용하기

- vi 작업 도중에 셸 명령을 실행하는 가장 간단한 방법은 ':! 셸 명령' 형태로 사용하는 것

```
(생략)
~
:! ls
```

```
[user1@localhost ch4]$ vi test.txt

exec.txt  exec2.txt  test.txt

계속하려면 엔터 혹은 명령을 입력하십시오
```

- test.txt 파일을 편집하는 중에 :! ls라고 하면 다음과 같은 형태로 실행
- vi를 빠져나가는 번거로움 없이 바로 이용할 수 있음
- 셸 명령의 결과를 확인하고 다시 vi 작업으로 돌아가려면 Enter 를 누름

02 vi 사용법

:sh 명령키 이용하기

- ':! 셸 명령'은 한 번에 하나의 셸 명령만 실행할 수 있음. 셸 명령이 여러 개 라면 매번 ':! 셸 명령'을 실행해야 하므로 불편함
- vi를 잠시 빠져나가서 셸 작업을 수행하고 다시 돌아오는 것이 편리

```
(생략)
~
:sh
```

- :sh로 vi를 잠시 빠져나오면 프롬프트가 나타남
- 여기서 필요한 셸 작업을 마치고 다시 vi로 돌아오려면 exit를 입력하면 됨

```
[user1@localhost ch4]$ ls
exec.txt  exec2.txt  test.txt
[user1@localhost ch4]$ exit
```

02 vi 사용법

■ 기타 명령키

- vi를 편리하게 사용하기 위해 알아두면 좋은 명령키

표 4-19 기타 명령키

명령키	기능
Ctrl+l(소문자 L)	현재 화면을 다시 출력한다.
Ctrl+g	현재 커서 위치의 행 번호를 마지막 행에 출력한다.
Shift+j(대문자 J)	현재 행과 아래 행을 연결하여 한 행으로 만든다.
. (마침표)	바로 직전에 했던 명령을 반복한다.
~ (물결표)	커서 위치의 글자를 대문자나 소문자로 바꾼다.

■ 화면 다시 출력하기

- vi 작업 도중에 시스템 메시지나 다른 사용자가 보낸 메시지가 출력되어 화면이 이상하게 보일 때 Ctrl+l(소문자 L) 명령키를 입력하면 메시지가 사라지고 원래 작업 중이던 내용이 다시 출력

02 vi 사용법

■ 행 연결하기

- 행을 연결할 때는 J 명령키를 사용함. 커서가 위치한 행과 다음 행을 하나의 행으로 만들어 줌

```
I like linu HANBIT linux
I like linu HANBIT linux
HANBIT HANBIT vi study
(생략)
```

- 첫 번째 행에서 J 명령키를 입력하면 두 번째 행과 합쳐짐
- 두 행이 합쳐질 때 그 사이에 공백문자가 삽입되며, 커서가 그 공백문자로 이동

```
I like linu HANBIT linux I like linu HANBIT linux
HANBIT HANBIT vi study
(생략)
```

02 vi 사용법

■ 이전 명령 반복하기

- .(마침표) 명령키는 바로 앞에 했던 명령을 반복적으로 수행
- .를 입력하면 다시 J 명령키를 실행한 것과 같은 결과가 나옴. 단, 커서 이동 키는 . 명령키를 입력해도 반복 수행되지 않음

```
I like linu HANBIT linux I like linu HANBIT linux HANBIT HANBIT vi study
(생략)
```

■ 대·소문자 전환하기

- ~ 명령키는 커서가 놓인 곳의 글자가 소문자이면 대문자로, 대문자이면 소문자로 바꿔 줌
- 앞의 예에서 커서를 x로 옮기고 ~ 명령키를 입력하면 x가 X로 바뀌고 커서가 오른쪽으로 한 칸 이동

```
I like linu HANBIT linux I like linu HANBIT linuX HANBIT HANBIT vi study
(생략)
```

02 vi 사용법

■ 기타 유용한 명령 익히기

① vi로 새로운 파일인 exec3.txt 파일을 연다

```
[user1@localhost ch4]$ vi exec3.txt
```

② i 키를 눌러 입력 모드로 전환하고 다음 내용을 입력

```
Good
morning
everyone.
```

③ Ctrl 를 입력하여 명령 모드로 전환하고 :w 명령으로 파일 내용을 저장

02 vi 사용법

■ 기타 유용한 명령 익히기

④ 3행 다음에 exec2.txt 파일을 읽어 들임

```
Good
morning
everyone.
Good afternoon everyone.        → :r exec2.txt Enter를 입력한다.
This is a living room.
My name is Gildong Park.
This is a living room.
My name is Gildong Hong.
Good afternoon everyone.
```

⑤ 1행과 2행을 연결하여 한 행으로 만듦

```
Good morning                    → ① 1G로 이동 → ② J 입력
everyone.
Good afternoon everyone.
(생략)
```

02 vi 사용법

■ 기타 유용한 명령 익히기

⑥ 반복 명령키로 1행과 2행을 연결하여 한 행으로 만듦

```
Good morning everyone          → .을 입력한다.
Good afternoon everyone.
(생략)
```

⑦ ~ 명령키로 1행의 everyone을 EVERYONE으로 바꿈

```
Good morning EVERYONE.         → ~~~~~~~~~를 입력한다.
Good afternoon everyone.
(생략)
```

02 vi 사용법

■ 기타 유용한 명령 익히기

⑧ exec3.txt 파일을 저장하고, 셸 명령을 실행하여 현재 디렉터리의 경로를 확인

```
Good morning EVERYONE.
Good afternoon everyone.
(생략)
~
:! pwd          ➞ ① :w Enter로 저장 → ② :! pwd Enter
```

```
[user1@localhost ch4]$ vi exec3.txt

/home/user1/Test/ch3                ➞ ① 경로 정보를 출력한다.

계속하려면 엔터 혹은 명령을 입력하십시오 ➞ ② Enter를 입력하여 vi로 복귀한다.
```

⑨ vi를 종료

02 vi 사용법

■ 기타 유용한 명령 익히기

혼자해보기 기타 유용한 명령 익히기

❶ vi로 exec3.txt 파일을 연다.
❷ 2행 다음에 exec.txt 파일을 읽어들인다.
❸ 3행을 삭제한다.
❹ 3행과 4행을 한 행으로 만든다.
❺ 파일을 저장하고 셸로 잠시 빠져나가 현재 디렉터리의 경로와 파일의 크기를 확인한다.
❻ 파일을 저장하고 종료한다.

03 vi 환경 설정

03 vi 환경 설정

vi 시작하기

- vi는 사용자가 환경을 설정할 수 있도록 set 명령을 제공
  - 사용자 홈 디렉터리에 .exrc 파일로 저장
  - 환경 변수 EXTINIT에 지정
  - vi의 마지막 행 모드에서 명령으로 설정

표 4-20 vi 환경 설정 명령

set 명령과 옵션	기능
set nu	파일 내용의 각 행에 행 번호를 표시한다(보이기만 할 뿐 저장되지는 않는다).
set nonu	행 번호를 감춘다.
set list	눈에 보이지 않는 특수문자를 표시한다(tab:^I, eol:$ 등).
set nolist	특수문자를 감춘다.
set showmode	현재 모드를 표시한다.
set noshowmode	현재 모드를 감춘다.
set	set으로 설정한 모든 vi 환경 설정값을 출력한다.
set all	모든 vi 환경 변수와 현재 값을 출력한다.

03 vi 환경 설정

■ 행 번호 표시

- :set nu를 입력하면 다음과 같이 행 번호가 표시

파일: exec.txt	set nu 실행 후
Good morning everyone. Nice to meet you. I am a linux expert. Now introduce yourself. ~	1 Good morning everyone. 2 Nice to meet you. 3 I am a linux expert. 4 Now introduce yourself. ~ :set nu

- 행 번호는 사용자의 편의를 위해 보이는 것으로 파일에 저장되지는 않음
- :set nonu를 입력하면 행 번호가 없어짐

03 vi 환경 설정

■ 특수 문자 표시

- 특수문자를 보려면 :set list 명령을 입력해야 함
- exec.txt 파일의 마지막에 탭을 하나 추가하고 :set list 명령을 실행한 예

파일: exec.txt	set list 실행 후
Good morning everyone. Nice to meet you. I am a linux expert. Now introduce yourself.〈탭〉 ~	Good morning everyone.$ Nice to meet you.$ I am a linux expert.$ Now introduce yourself.^I$ ~ :set list

- exec.txt 파일에서 :set list를 입력하면 특수문자 $와 ^I가 보임
- $는 행의 끝을 표시하고 ^I(Ctrl +대문자 i)는 탭을 나타냄

03 vi 환경 설정

■ 환경 설정값 표시

- :set 명령은 현재 사용 중인 vi의 환경 설정 상태를 보여줌

```
~
:set
--- 옵션 ---
  commentstring=      incsearch           scroll=16           ttyfast
  display=truncate    langnoremap         scrolloff=5         ttymouse=sgr
  filetype=text       nolangremap         showcmd             viminfo='20,"50
  helplang=ko         modified            syntax=text         wildmenu
  history=200         nrformats=bin,hex   ttimeout            t_Sb=^[[4%dm
  hlsearch            ruler               ttimeoutlen=100     t_Sf=^[[3%dm
  backspace=indent,eol,start
  comments=fb:-,fb:*,n:>
  fileencoding=utf-8
  fileencodings=ucs-bom,utf-8,default,latin1
  guicursor=n-v-c:block,o:hor50,i-ci:hor15,r-cr:hor30,sm:block,a:blinkon0
계속하려면 엔터 혹은 명령을 입력하십시오
```

03 vi 환경 설정

■ 모든 환경 변수 표시

- vi의 환경 설정에 사용되는 모든 환경 변수와 현재 설정값을 보여주는 명령은 :set all

```
:set all
--- 옵션 ---
   aleph=224          fixendofline       nonumber           synmaxcol=3000
   noarabic           foldclose=         numberwidth=4      syntax=text
   arabicshape        foldcolumn=0       omnifunc=          tabline=
   noallowrevins      foldenable         operatorfunc=      tabpagemax=10
   ambiwidth=single   foldexpr=0         nopaste            tabstop=8
(생략)
   nocindent          iminsert=0         remap              ttimeoutlen=100
-- 더 --
```

03 vi 환경 설정

.exrc 파일에 환경 설정하기

- vi 환경 설정은 사용자 홈 디렉터리에 .exrc 파일로 저장해 놓을 수 있음. 이 파일은 원래 존재하는 파일이 아니므로 사용자가 만들어야 함
- .exrc 파일에 환경을 설정할 때는 다음과 같이 set 명령과 옵션만 지정하면 됨

```
set nu
set list
set showmode
```

03 vi 환경 설정

EXINIT 환경 변수에 설정하기

- vi 환경 설정은 다음과 같이 셸의 환경 변수인 EXINIT에도 가능

```
[user1@localhost ch4]$ EXINIT='set nu list'
[user1@localhost ch4]$ export EXINIT
```

- 환경 변수 EXINIT의 설정을 삭제하기

```
[user1@localhost ch4]$ EXINIT=''
[user1@localhost ch4]$ export EXINIT
```

Thank you!

한빛아카데미 Hanbit Academy, Inc.