Shell, vim command for git

Fastcampus Programming SCHOOL

최우영

- Co-founder, CTO(disceptio)
- Solution Architect, Web Developer, Instructor
- Skills: Python, Golang, Julia, Node.js, Google tag manager ...

Contacts

1. blog: https://ulgoon.github.io/
2. github: https://github.com/ulgoon/
3. email: me@ulgoon.com
4. discord: @ulgoon

Introduce

Goal(1)

- Linux의 역사를 이해한다
- CLI에 대한 공포를 극복하고 Shell과 친구가 된다
- Linux Shell 커맨드를 학습하여 능숙하게 이를 활용할 수 있다
- Vim 텍스트 에디터를 이용하여 커밋메시지를 작성하고, 파일 수정을 할 수 있다

Goal(2)

- git을 이해하고, git과 github이 다름을 인지한다
- git을 활용하여 나의 소스코드를 관리할 수 있다
- 나의 커리어를 Swag 할 수 있는 블로그를 git을 활용하여 관리할 수 있다.
- git의 branch model을 활용해 능숙하게 코드관리할 수 있다
- git으로 타인과 협업하며, 다른 프로젝트에 기여할 수 있다

Contents

- Introduction to Linux, Shell
- Shell commands
- Vim commands

Before Linux(1)

- 1965년 데니스 리치, 켄 톰슨 외 x명이 AT&T Bell 연구소에서 PDP-7 기반 어셈블리어로 작성한 UNIX를 개발

Before Linux(2)

- 1973년 데니스 리치와 켄 톰슨이 C를 개발한 뒤, C 기반 UNIX 재작성

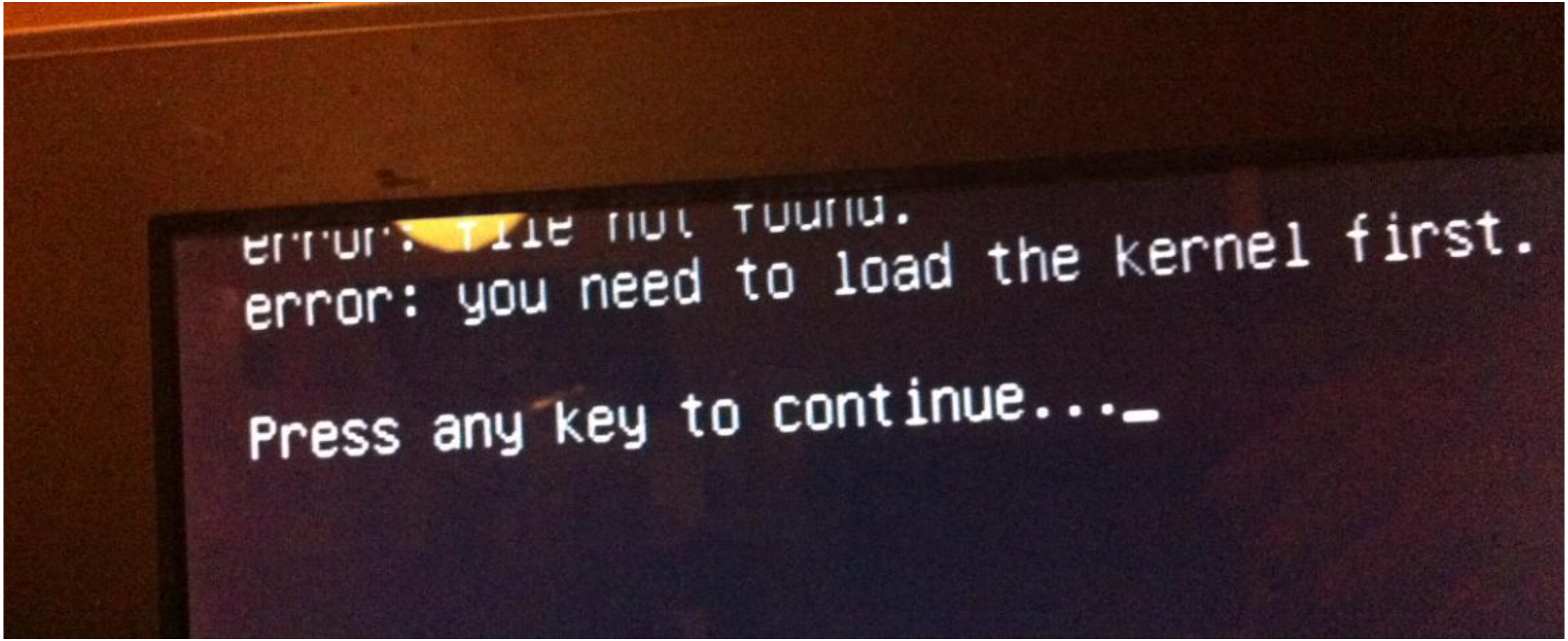
Before Linux(3)

- 1984년 리차드 스톨먼이 오픈 소프트웨어 자유성 확보를 위한 GNU 프로젝트 돌입

Meaning of GNU

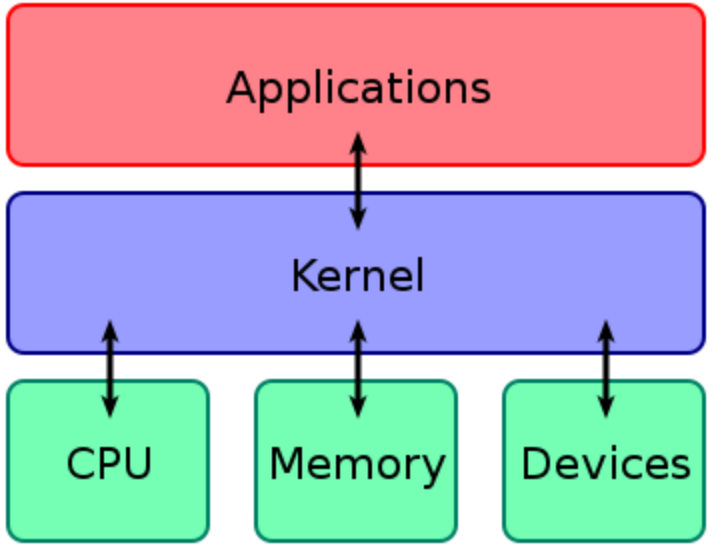
GNU == `G` NU is `N` ot `U` nix

Before Linux(4)



- But, GNU 프로젝트에는 커널이 없었고..

Kernel



- 하드웨어와 응용프로그램을 이어주는 운영체제의 핵심 시스템소프트웨어

Linus Torvalds

https://www.youtube.com/embed/IVpOyKCNZYw

- 헬싱키 대학생이던 리누스 토발즈는 앤디 타넨바움의 MINIX를 개조한 Linux를 발표
- 0.1 - bash(GNU Bourne Again SHell), gcc(UNIX 기반 C 컴파일러)

Linux

- 리누스 토발즈가 작성한 커널 혹은 GNU 프로젝트의 라이브러리와 도구가 포함된 운영 체제
- PC와 모바일, 서버, 임베디드 시스템 등 다양한 분야에서 활용
- Redhat, Debian, Ubuntu, Android 등 다양한 배포판이 존재

Shell

- 운영체제의 커널과 사용자를 이어주는 소프트웨어
- sh(Bourne Shell): AT&T Bell 연구소의 Steve Bourne이 작성한 유닉스 쉘
- csh: 버클리의 Bill Joy가 작성한 유닉스 쉘(C언어랑 비슷한 모양)
- bash(Bourne Again Shell): Brian Fox가 작성한 유닉스 쉘
    - 다양한 운영체제에서 기본 쉘로 채택
- zsh: Paul Falstad가 작성한 유닉스 쉘
    - sh 확장형 쉘
    - 현재까지 가장 완벽한 쉘

Let's learn bash

Shell Command Basic(1)

```
$ cd documents

$ mkdir dev # - make directory dev
$ cd dev # - change directory
$ cd .. # - go up
$ pwd # - print working directory

$ ls
$ ls -al

$ touch hello.py # - create hello.py
$ exit # - terminate shell
```

chmod

> 파일의 권한을 설정할 때 사용

`drwxr-xr-x`

`d` or `-` : directory or file

(user)(group)(other)

`r` : read

`w` : write

`x` : execute

`-` : no permission

chmod

```
$ chmod [옵션] (8진수) (파일명)
```

8진수

0: 000

1: 001

2: 010

3: 011

4: 100

5: 101

6: 110

7: 111

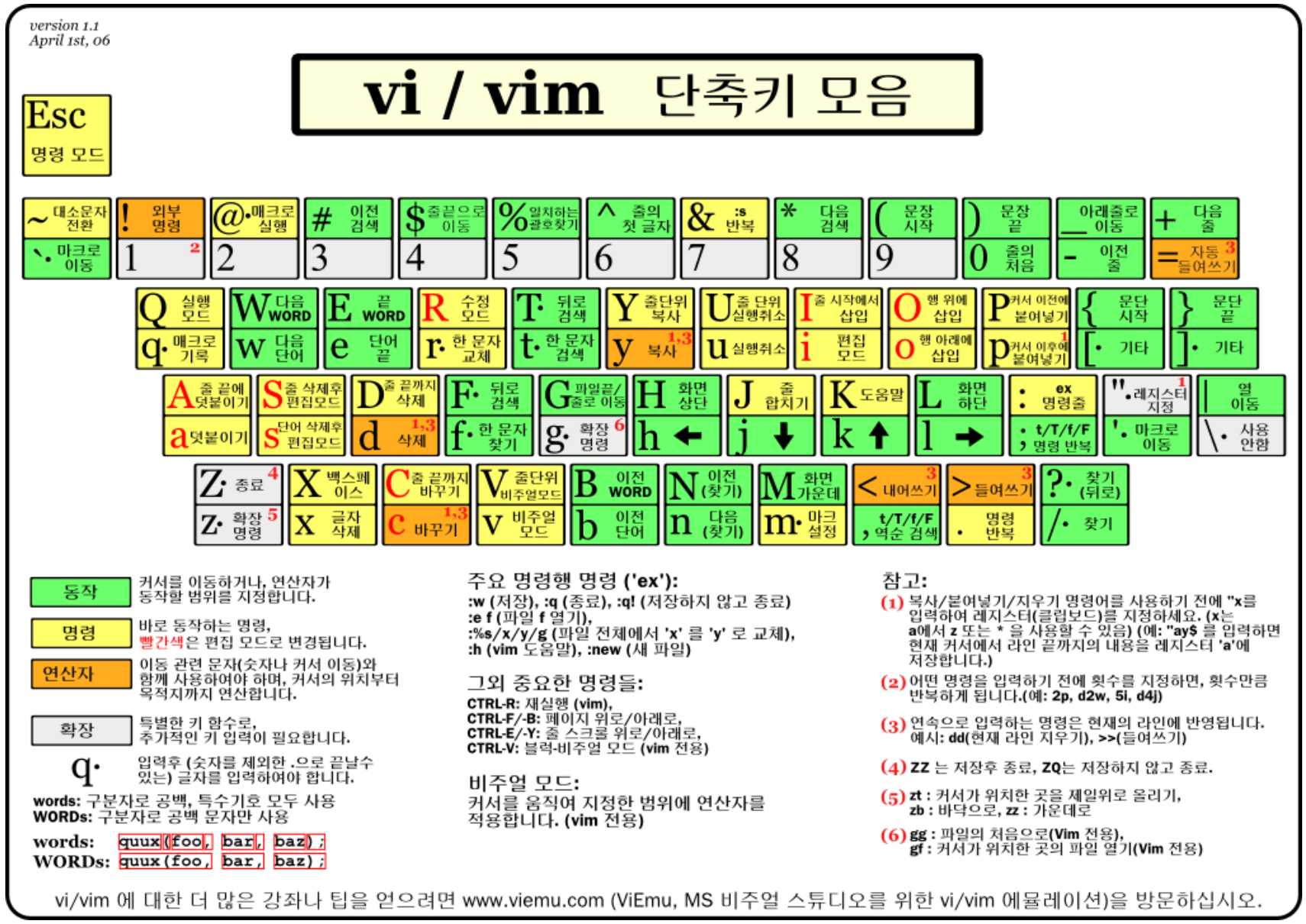
Shell Command Basic(2)

```
$ mv hello.py dev
$ cp hello.py dev

$ rm hello.py
$ rm -rf dev/

$ cat README.md
$ head README.md
$ tail -20 README.md
$ cat README.md > README.txt
$ open README.md # use explorer instead open on windows
```

Vim

TODO

1. documents/dev에서 fastcampus 디렉토리를 만듭니다
2. 새파일 index.html을 만들어주세요
3. vim으로 내부 작업을 한뒤 저장하여 cat 명령어로 저장을 확인합니다.
4. index.html을 style.css로 이름을 변경합니다.


version 1.1
April 1st, 06
vi / vim 단축키 모음
Esc
명령 모드
~ 대소문자 전환
` 마크로 이동
! 외부 명령
1
@ 매크로 실행
2
이전 검색
3
$ 줄끝으로 이동
4
% 일치하는 괄호찾기
5
^ 줄의 첫 글자
6
& :s 반복
7
* 다음 검색
8
(문장 시작
9
) 문장 끝
0 줄의 처음
_ 아래줄로 이동
- 이전 줄
+ 다음 줄
= 자동 들여쓰기
Q 실행 모드
q 매크로 기록
W 다음 WORD
w 다음 단어
E 끝 WORD
e 단어 끝
R 수정 모드
r 한 문자 교체
T 뒤로 검색
t 한 문자 검색
Y 줄단위 복사
y 복사
U 줄 단위 실행취소
u 실행취소
I 줄 시작에서 삽입
i 편집 모드
O 행 위에 삽입
o 행 아래에 삽입
P 커서 이전에 붙여넣기
p 커서 이후에 붙여넣기
{ 문단 시작
[기타
} 문단 끝
] 기타
A 줄 끝에 덧붙이기
a 덧붙이기
S 줄 삭제후 편집모드
s 단어 삭제후 편집모드
D 줄 끝까지 삭제
d 삭제
F 뒤로 검색
f 한 문자 찾기
G 파일끝/줄로 이동
g 확장 명령
H 화면 상단
h
J 줄 합치기
j
K 도움말
k
L 화면 하단
l
: ex 명령줄
; t/T/f/F 명령 반복
" 레지스터 지정
' 마크로 이동
| 열 이동
\ 사용 안함
Z 종료
z 확장 명령
X 백스페이스
x 글자 삭제
C 줄 끝까지 바꾸기
c 바꾸기
V 줄단위 비주얼모드
v 비주얼 모드
B 이전 WORD
b 이전 단어
N 이전 (찾기)
n 다음 (찾기)
M 화면 가운데
m 마크 설정
< 내어쓰기
t/T/f/F , 역순 검색
> 들여쓰기
. 명령 반복
? 찾기 (뒤로)
/ 찾기
동작
커서를 이동하거나, 연산자가 동작할 범위를 지정합니다.
명령
바로 동작하는 명령, 빨간색은 편집 모드로 변경됩니다.
연산자
이동 관련 문자(숫자나 커서 이동)와 함께 사용하여야 하며, 커서의 위치부터 목적지까지 연산합니다.
확장
특별한 키 함수로, 추가적인 키 입력이 필요합니다.
q
입력후 (숫자를 제외한 .으로 끝날수 있는) 글자를 입력하여야 합니다.
words: 구분자로 공백, 특수기호 모두 사용
WORDs: 구분자로 공백 문자만 사용
words: quux(foo, bar, baz);
WORDs: quux(foo, bar, baz);
주요 명령행 명령 ('ex'):
:w (저장), :q (종료), :q! (저장하지 않고 종료)
:e f (파일 f 열기),
:%s/x/y/g (파일 전체에서 'x' 를 'y' 로 교체),
:h (vim 도움말), :new (새 파일)
그외 중요한 명령들:
CTRL-R: 재실행 (vim),
CTRL-F/-B: 페이지 위로/아래로,
CTRL-E/-Y: 줄 스크롤 위로/아래로,
CTRL-V: 블럭-비주얼 모드 (vim 전용)
비주얼 모드:
커서를 움직여 지정한 범위에 연산자를 적용합니다. (vim 전용)
참고:
(1) 복사/붙여넣기/지우기 명령어를 사용하기 전에 "x를 입력하여 레지스터(클립보드)를 지정하세요. (x는 a에서 z 또는 * 을 사용할 수 있음) (예: "ay$ 를 입력하면 현재 커서에서 라인 끝까지의 내용을 레지스터 'a'에 저장합니다.)
(2) 어떤 명령을 입력하기 전에 횟수를 지정하면, 횟수만큼 반복하게 됩니다.(예: 2p, d2w, 5i, d4j)
(3) 연속으로 입력하는 명령은 현재의 라인에 반영됩니다. 예시: dd(현재 라인 지우기), >>(들여쓰기)
(4) ZZ 는 저장후 종료, ZQ는 저장하지 않고 종료.
(5) zt : 커서가 위치한 곳을 제일위로 올리기, zb : 바닥으로, zz : 가운데로
(6) gg : 파일의 처음으로(Vim 전용), gf : 커서가 위치한 곳의 파일 열기(Vim 전용)
vi/vim 에 대한 더 많은 강좌나 팁을 얻으려면 www.viemu.com (ViEmu, MS 비주얼 스튜디오를 위한 vi/vim 에뮬레이션)을 방문하십시오.

Vim Basic

`h,j,k,l – move cursor`

`i – insert mode`

`v – visual mode`

`d – delete`

`y – yank`

`p – paste`

`u – undo`

`r – replace`

`$ – move end of line`

`^ – move start of line`

`:q – quit`

`:q! – quit w/o write(no warning)`

`:wq – write and quit`

`:{number} – move to {number}th line`

write `hello.py` with Vim

`$ vim`

`$ vim hello.py`

`i`

`-- insert --`

type `print("hello python!")`

press `esc` to escape

`:wq`

`$ python hello.py`

copy & paste

`$ vim hello.py`

`v`

`-- visual --`

블록지정 후 `y`

`p`

press `esc` to escape

`:wq`

`$ python hello.py`

Use Vim in real world!

- vim advanture
- vimium
- vs code vim