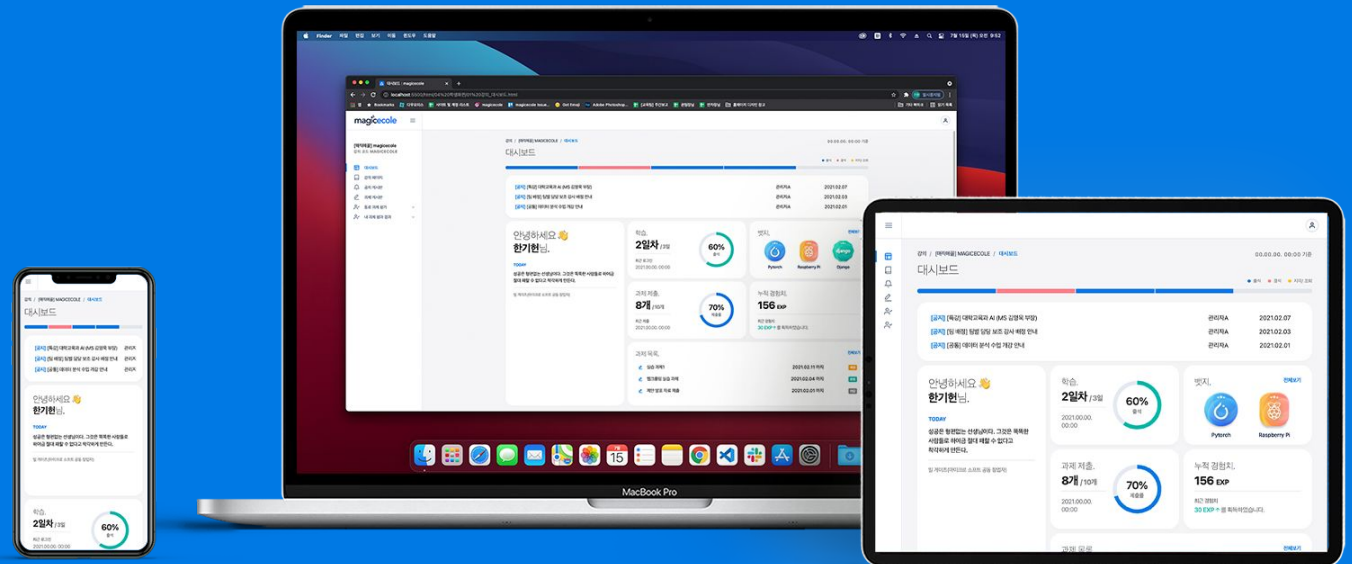


DevOps - Linux

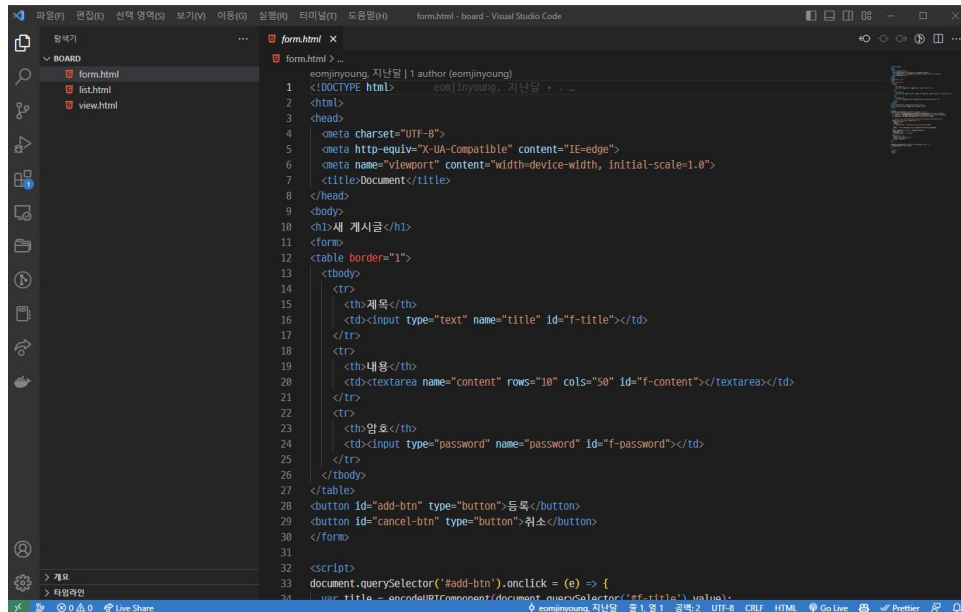
텍스트 파일 편집



다룰 내용

- vi와 nano 비교
- nano 기본 사용법

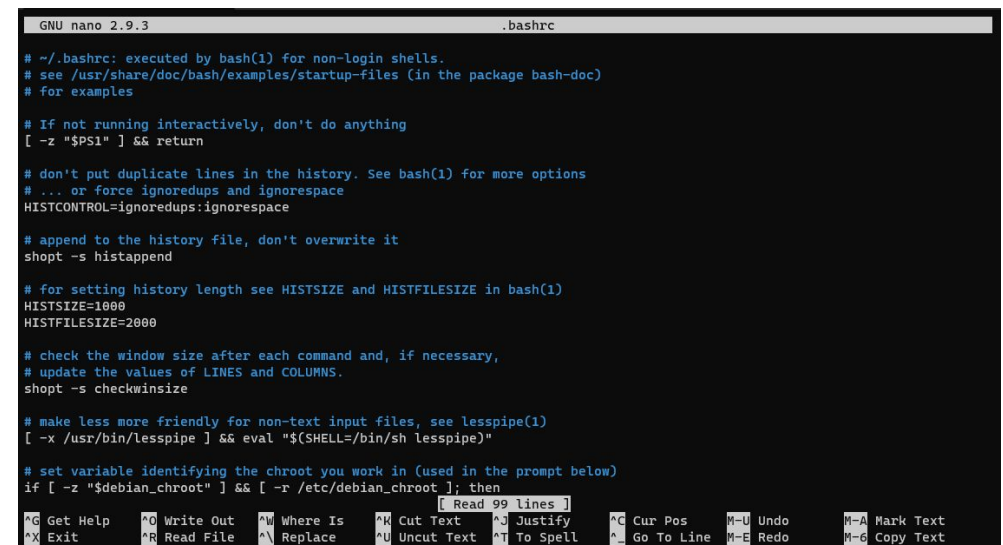
윈도우(또는 MacOS) vs 리눅스

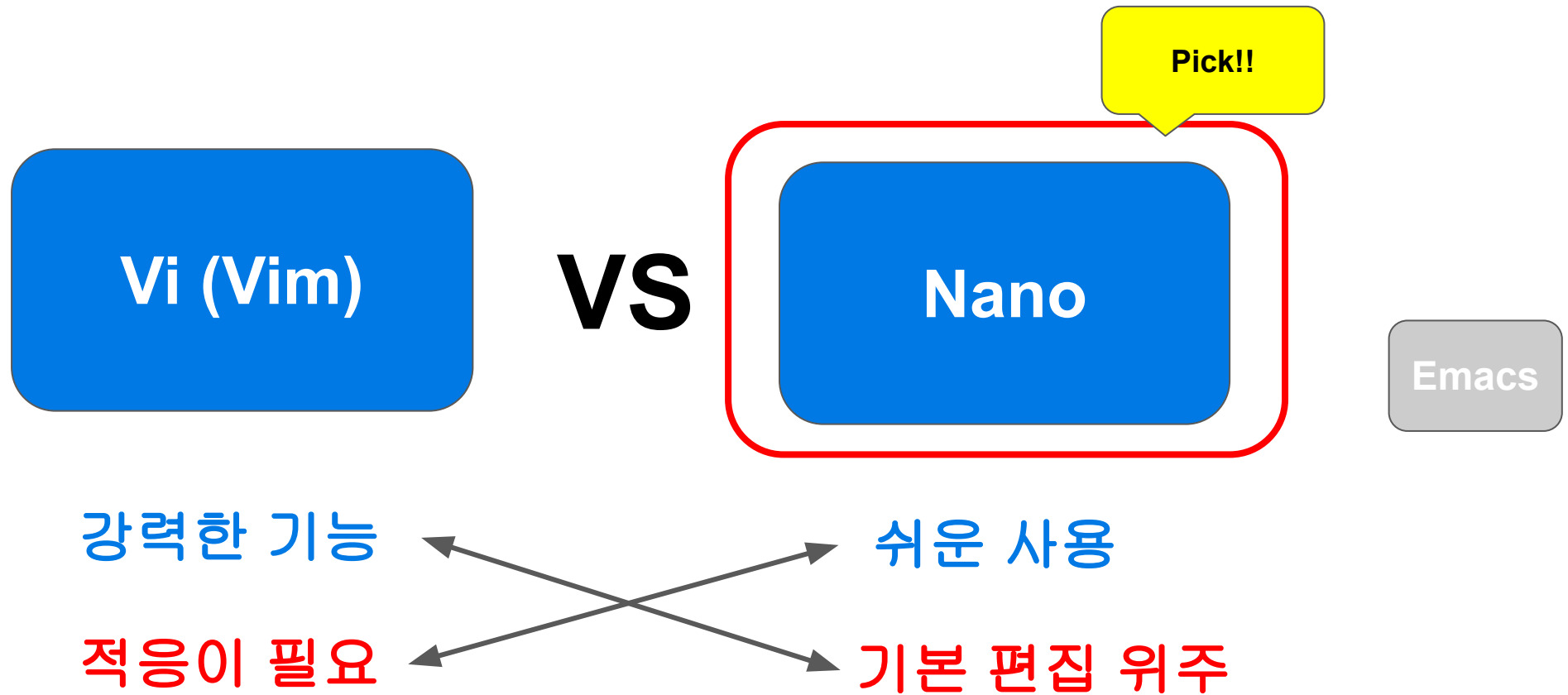


윈도우 환경
- With Mouse -

VS

리눅스 환경
- Only Keyboard -





Only Keyboard - 손에 익혀야 함!!

nano 실행

- nano

- **형식** : nano **[파일]**

1. nano 에디터만 실행 (빈 문서) : nano

```
root@linux-test:~# nano
```

GNU nano 2.9.3

New Buffer

편집창 (편집 버퍼)

단축키 표시

^G Get Help	^O Write Out	^W Where Is	^K Cut Text	^J Justify	^C Cur Pos
^X Exit	^R Read File	^_ Replace	^U Paste Text	^T To Spell	^_ Go To Line

종료 : Ctrl + x
(또는 F2 키)

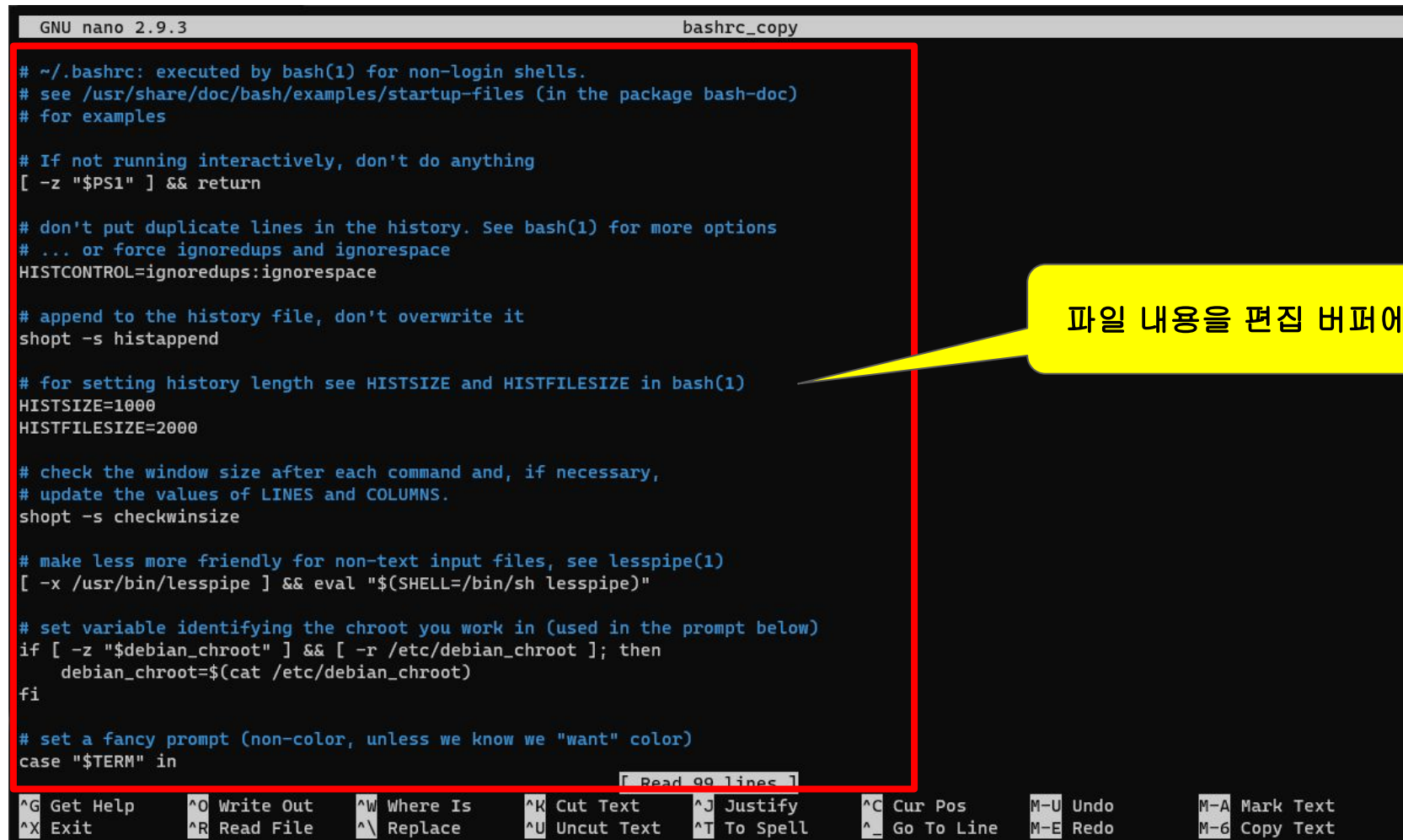
^G Get Help	^O Write Out	^W Where Is	^K Cut Text	^J Justify	^C Cur Pos
^X Exit	^R Read File	^_ Replace	^U Uncut Text	^T To Spell	^_ Go To Line

터미널 창 크기를 늘리면
단축키도 많이 표시됨

2. nano 에디터로 bashrc_copy 파일 불러오기 : **nano bashrc_copy**

```
root@linux-test:~# cp .bashrc bashrc_copy
root@linux-test:~# nano bashrc_copy
```

연습 파일
복사



```
GNU nano 2.9.3 bashrc_copy
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
[ -z "$PS1" ] && return

# don't put duplicate lines in the history. See bash(1) for more options
# ... or force ignoredups and ignorespace
HISTCONTROL=ignoredups:ignorespace

# append to the history file, don't overwrite it
shopt -s histappend

# for setting history length see HISTSIZE and HISTFILESIZE in bash(1)
HISTSIZE=1000
HISTFILESIZE=2000

# check the window size after each command and, if necessary,
# update the values of LINES and COLUMNS.
shopt -s checkwinsize

# make less more friendly for non-text input files, see lesspipe(1)
[ -x /usr/bin/lesspipe ] && eval "$(SHELL=/bin/sh lesspipe)"

# set variable identifying the chroot you work in (used in the prompt below)
if [ -z "$debian_chroot" ] && [ -r /etc/debian_chroot ]; then
    debian_chroot=$(cat /etc/debian_chroot)
fi

# set a fancy prompt (non-color, unless we know we "want" color)
case "$TERM" in
```

[Read 99 lines]

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos M-U Undo M-A Mark Text
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line M-E Redo M-6 Copy Text

파일 내용을 편집 버퍼에 읽어옴

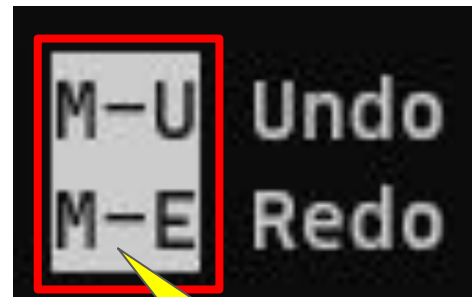
nano 기본 조작

커서 이동 : 상/하/좌/우 키

단축키 : **Ctrl / Alt + [키]** 조합



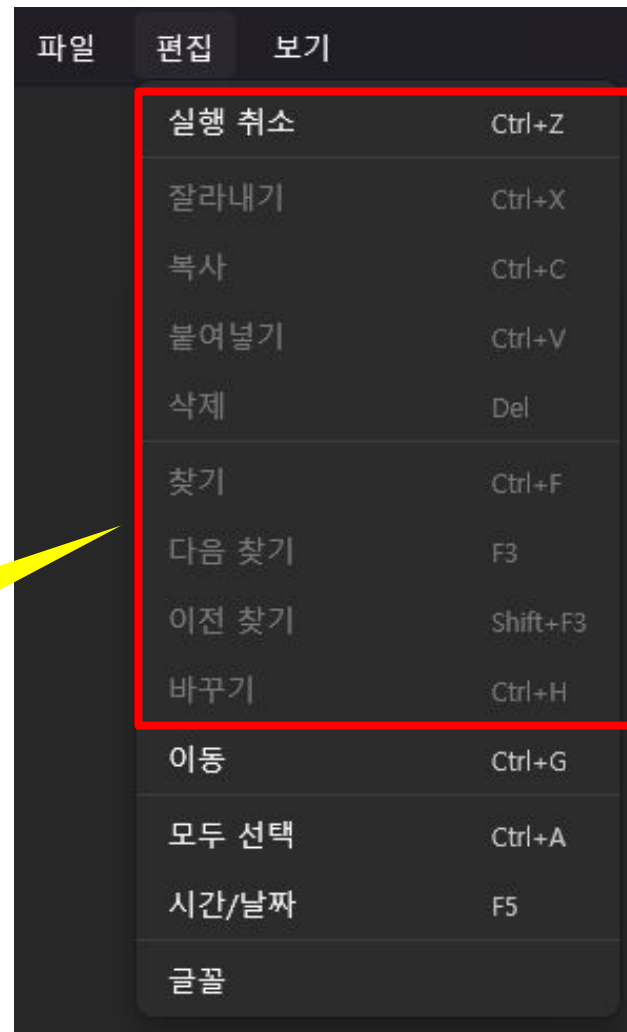
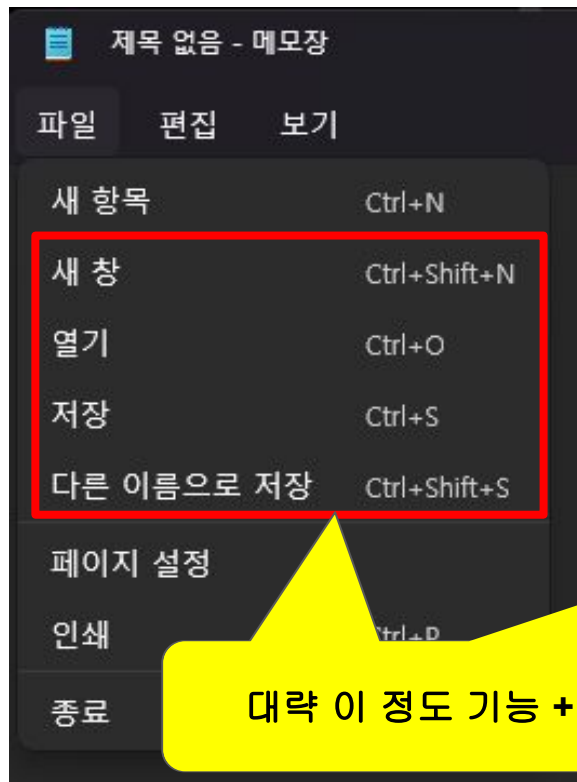
Ctrl + g
Ctrl + x



Alt + u
Alt + e

nano 기본 사용법

참고 : 윈도우 메모장 기능



nano 기본 사용법

도움말

저장

찾기

잘라내기

텍스트 선택되지 않으면
한 줄 잘라내기

^G Get Help
^X Exit

^O Write Out
^R Read File

^W Where Is
^_ Replace

^K Cut Text
^U Uncut Text

열기 (파일 읽어오기)

새로 읽기 또는 현재 파일에 추가

바꾸기

붙여 넣기

실행 취소

텍스트 선택
(Shift + 방향키)

이전 찾기

줄 처음으로
이동

M-U Undo
M-E Redo

M-A Mark Text
M-6 Copy Text

M-] To Bracket
M-W WhereIs Next

M-^ Previous
M-^ Next

^B Back
^F Forward

^_ Prev Word
^_ Next Word

^A Home
^E End

다시 실행

복사

텍스트 선택되지 않으면
한 줄 복사

다음 찾기

단어 단위
이동

줄 끝으로
이동

nano 기본 사용법

기능에 따른 추가 입력

^O Write Out

저장

File Name to Write:

^G Get Help **M-D** DOS Format **M-A** Append **M-B** Backup File
^C Cancel **M-M** Mac Format **M-P** Prepend **^T** To Files

편집 버퍼의 내용을 저장할 파일 경로 입력

^R Read File

열기

File to insert [from ./]:

^G Get Help **M-F** New Buffer **^X** Execute Command
^C Cancel **M-N** No Conversion **^T** To Files

현재 편집 버퍼에 추가할(Append) 파일 경로 입력

M-F : 새로운(New) 편집 버퍼에 읽어오기

^W Where Is

찾기

Search:

^G Get Help **M-C** Case Sens **M-B** Backwards **^P** Older **^T** Go To Line
^C Cancel **M-R** Regexp **^R** Replace **^N** Newer **M-J** FullJustify

현재 편집 버퍼에서 검색할 문자열 입력

^_ Replace

바꾸기

Search (to replace):

^G Get Help **M-C** Case Sens **M-B** Backwards **^P** Older
^C Cancel **M-R** Regexp **^R** No Replace **^N** Newer

현재 편집 버퍼에서 검색할 문자열 입력

Replace with:

^G Get Help **^P** Older
^C Cancel **^N** Newer

바꾸 입력할 문자열 입력

도움말 참고 : Ctrl + g

```
^G      (F1)      Display this help text
^X      (F2)      Close the current buffer / Exit from nano
^O      (F3)      Write the current buffer (or the marked region) to disk
^R      (Ins)     Insert another file into current buffer (or into new buffer)

^W      (F6)      Search forward for a string or a regular expression
^_      (M-R)     Replace a string or a regular expression
^K      (F9)      Cut current line (or marked region) and store it in cutbuffer
^U      (F10)     Paste the contents of cutbuffer at current cursor position

^J      (F4)      Justify the current paragraph
^T      (F12)     Invoke the spell checker, if available

^C      (F11)     Display the position of the cursor
^_      (M-G)     Go to line and column number

M-U      Undo the last operation
M-E      Redo the last undone operation

M-A      (^6)     Mark text starting from the cursor position
M-6      (M-^)    Copy current line (or marked region) and store it in cutbuffer

M-]      Go to the matching bracket

^Q      Search backward for a string or a regular expression
M-Q      Search next occurrence backward
M-W      Search next occurrence forward

^B      (◀)       Go back one character
^F      (▶)       Go forward one character
^◀      (M-Space) Go back one word
^▶      (^Space)  Go forward one word
^A      (Home)    Go to beginning of current line
^E      (End)     Go to end of current line
```

Vim 에디터

Vim 셋팅을 자알하면....

```
.. (up a dir)
</Mobibench/shell/sqlite_3.7.5/
sqlite3.c
sqlite3.h

14869 mem.xBacktrace(pHdr->iSize, pHdr->nBacktrace-1, &pBt[1]);
14870 }
14871 }
14872
14873 /*
14874 ** Open the file indicated and write a log of all unfreed memory
14875 ** allocations into that log.
14876 */
14877 SQLITE_PRIVATE void sqlite3Memdebug(
14878     FILE *out;
14879     struct MemBlockHdr *pHdr;
14880     void **pBt;
14881     int i;
14882     out = fopen(zFilename, "w");
14883     if( out==0 ){
14884         fprintf(stderr, "** Unable to open memory debug output log: %s **\n",
14885             zFilename);
14886         return;
14887     }
14888     for(pHdr=mem.pFirst; pHdr; pHdr=pHdr->pNext){
14889         char *z = (char*)pHdr;
14890         z -= pHdr->nBacktraceSlots*sizeof(void*) * pHdr->nTitle;
14891         fprintf(out, "**** %ld bytes at %p from %s ****\n",
14892             pHdr->iSize, &pHdr[1], pHdr->nTitle ? z : "???");
14893         if( pHdr->nBacktrace ){
14894             fflush(out);
14895             pBt = (void**)pHdr;
14896             pBt -= pHdr->nBacktraceSlots;
14897             backtrace_symbols_fd(pBt, pHdr->nBacktrace, fileno(out));
14898             fprintf(out, "\n");
14899         }
14900     }
14901 }

sqlite3.c
7533 };
7534
7535
7536 SQLITE_PRIVATE int sqlite3BtreeOpen(
7537     const char *zFilename, /* Name of database file to open */
7538     sqlite3 *db, /* Associate
7539     Btree **ppBtree, /* Return op
7540     int flags, /* Flags */
7541     int vfsFlags /* Flags pas

</Mobibench/shell/sqlite_3.7.5 sqlite3.c
sqlite3.c, filetype=c, Tag count=4744
```

파일 탐색

화면 분할

함수/변수
리스트

함수/변수의 정의/호출 탐색

관심 있으면 찾아보세요^^

```
Cscope tag: start_kernel
# line filename / context / line
1 134 arch/alpha/boot/bootp.c <<start_kernel>>
start_kernel(void )
2 262 arch/alpha/boot/bootpz.c <<start_kernel>>
start_kernel(void )
3 151 arch/alpha/boot/main.c <<start_kernel>>
void start_kernel(void )
4 480 init/main.c <<start_kernel>>
asmlinkage __visible void __init start_kernel(void )

Type number and <Enter> (empty cancels):
```

Thank You