

vim

```
Vim??vi????????????????????????????????????????????????????????Emacs?????Unix?
???????????? Vim?????????·???1991????
```

- [Vim ????](#)
- [Vim FAQ](#)
- [vimrc](#)
- Cheat Sheets

Vim ?????

????

???????????????	????	????
i	???????????????	? a ????????????????????????????
a	???????????????	
o	?????????????	
h j k l	?????????h--??, j--??, k--??, l--??	?????????????????????????????????
:1 : 22	???????? ??????22?	?????????????????
gg	???????	
G	???????	???????????????????????????
0 / \$????????/??	
w / W	????????????/??	?????
b / B	????????????/??	?????
x	???????	
J (Shift + j)	???????????	?????????
{	???????????	
}	???????????	
:wq	???????	

:w new.txt	????? new.txt	
:w !sudo tee %	? sudo ?????	???????????????? sudo ???????
:q!	?????	???????????????????????????????? ?
:set nu!	??????	
:set hlsearch	Highlight ??????	???????????
:set cuc!	??????	?? json, yaml ??
: syntax on	?? Hihjlight ??	
d dd	??????? ????????	
10dd	???????10?(???)	
y yy	??????? ????????	
3yy	???????3?(???)	
p	??	
/keyword	????	
/KeyWord	????(?????)	
/S[Uu]SE	????????????	
*	????????????	
:%s/old/new/g :%s/old/new/gc	???????old????????new?? gc: ?????	
:s#/usr/local/bin#/usr/sbin#g	? # ??????	
:10,30 s/old/new/g	?10-30??? old ? new	

:%s/old1\ OLD2/new/gc	???? old1?OLD2 ? new	
:/<table>/,/</table>/g/^\$/d	?? <table> ... </table> ?????	
:%s\t/ /g	?? Tab ?4???	??????
/s\+\$??????????? Highlight	????? :set hlsearch ??, not support AIX
:%s\s\+\$//	????????????	not support AIX
~	?????	
:%s/[ctrl+V and ctrl+M]//g	?????? ^M	
ggVG	????	?? Ctrl a
dG	????????????	
dgg	????????????	
:1,\$d	??????	
:1,22d	?? 1-22 ?	
:23,\$d	?? 23 ???????	
:! ls -l	???? SHELL ??	:!! ?????
:vsplit my.file	???????????? my.file	ctrl + w + w ????
:split my.file	??????	ctrl + w + w ????
:sort :sort! :sort u :sort i :sort n	?? ???? ????????? ????????? ??????	
u Ctrl + r	Undo Redo	

- `??????? :set hlsearch`
- `??????????? :set incsearch`
- `?? {,},[]() ?????? %`
- `???????????????? { ? [{`
- `????????????????????? zt (Top), zz (Middle), zb (Bottom)`
- `????????????????? H (Top), M (Middle), L (Last Line)`
- `????? Ctrl+b (???), Ctrl+f (???)`
- `????? Ctrl+y (??), Ctrl+e (??)`

??????

- ????????? /old -> cw -> ?????? new -> ?????? Esc -> ?????? n -> ?????? .
- ???????
 1. ???shift + v
 2. ????
 3. ?????? :s/old/new/g (?????????????)

????

[] p []

```
:<[ ]>yank
```

[16 boxes]. [2 boxes] copy [1 box] t

:<□□>copy.

$$:\langle \Box \rangle t.$$

10

$$: + 10t.$$

□□□□□□□□ 10 □□□□□□□□□□

$$:-10t.$$

□□□□□□(33 ~ 44)□□□□

:33,44t.

????????????????

1. Ctrl + v
2. ?????????
3. ??????
4. Shift + I
5. ?????????
6. Esc ?????

?????????????`;

1. ?????????
2. Ctrl + **v**
3. ????????
4. Shift + **\$**
5. Shift + **A**
6. ????????? **;**
7. Esc ?????

????

1. Shift + **v**
2. ????
3. **:norm i#** (??????)
4. **:norm x** (??????)

????

1. Shift + **v** (Visually Seleting)
2. ????
3. **:w /path/to/newfile** (Save as newfile)
4. **gv** (reselect the last visual selection)
5. **d** (delete that selection)

?????? (auto-complete)

1. ??????
2. ?? /etc/profile??????
3. Ctrl + **x** -> Ctrl **f**

Tutorials

- [Youtube Channel - Vim \(TheFrugalComputerGuy\)](#)
- [VIM Colors Online](#)
- [VimTricks](#)

Vim FAQ

Err: A line cannot be longer than 2048 characters.

???? VI ??????????? 2048 ?????????????????????????

```
cut -c1-2047 before.log > after.log
```

?? 3X MB ????????

“ ex: 0602-101 Out of memory saving lines for undo.

the number must be less than 8 digits.

```
vi -y 99999999 your-file
```

???????

?????????????(autoindent)??????????????

Solution: ???????? `:set paste`

0602-140 The window is too small to display the current line.

AIX ????????????????????

Solution:

1. ? putty ???????????????? 9 ???
2. ?? Vim ?? Vi?

Unable to paste from clipboard on AIX

? Windows ???????? putty ??????????

Solution: `:set mouse=v`

vimrc

vimrc simple

```
"
" Fixed for Python virtualenv as the follow error.
" ModuleNotFoundError: No module named 'powerline'
"

python3 import sys;sys.path.append("/home/along/.local/lib/python3.10/site-packages")
"

" Powerline with Vim
set laststatus=2
python3 from powerline.vim import setup as powerline_setup
python3 powerline_setup()
python3 del powerline_setup

"

syntax on          " Enable syntax highlighting
set number         " Show line numbers
set relativenumber " Show relative line numbers
set tabstop=4      " Set tab width to 4 spaces
set shiftwidth=4   " Set indentation width to 4 spaces
set expandtab       " Use spaces instead of tabs
set cursorline     " Highlight the current line

" Moving Lines
" Ctrl-j: Move down, Ctrl-k: Move down
nnoremap <c-j> :m .+1<CR>==
nnoremap <c-k> :m .-2<CR>==
inoremap <c-j> <Esc>:m .+1<CR>==gi
inoremap <c-k> <Esc>:m .-2<CR>==gi
vnoremap <c-j> :m '>+1<CR>gv=gv
vnoremap <c-k> :m '<-2<CR>gv=gv

" Preserve last editing position
silent! source $VIMRUNTIME/defaults.vim
```

```
" Copy to Clipboard
" Prerequisites: apt install vim-gtk3
" Usage: Press v to select the lines, and then press y to copy
set clipboard=unnamedplus
```

vimrc advanced

```
"
" minimal vimrc with no (extra) plugins
"

"regular settings
"-----

" ui
set number
set ruler
set wildmenu
set showcmd
set showmatch

" encoding/format
set encoding=utf-8
set fileformats=unix,dos,mac

" searching
set hlsearch
set incsearch
set ignorecase
set smartcase

" indent
set shiftwidth=4
set tabstop=4
set softtabstop=4
set autoindent

" allow syntax and filetype plugins
syntax enable
filetype plugin indent on
```

runtime macros/matchit.vim

" Moving Lines

" Ctrl-j: Move down, Ctrl-k: Move up

nnoremap <c-j> :m .+1<CR>==

nnoremap <c-k> :m .-2<CR>==

inoremap <c-j> <Esc>:m .+1<CR>==gi

inoremap <c-k> <Esc>:m .-2<CR>==gi

vnoremap <c-j> :m '>+1<CR>gv=gv

vnoremap <c-k> :m '<-2<CR>gv=gv

" autocmds

"-----

augroup general

autocmd!

"keep equal proportions when windows resized

autocmd VimResized * wincmd =

"save cursor position in a file

autocmd BufReadPost * if line("\") > 1 && line("\")

\ <= line("\$") | exe "normal! g\"" | endif

augroup END

augroup languages

autocmd!

autocmd BufNewFile,BufRead *.bash set syntax=sh

autocmd FileType python xnoremap <leader>r <esc>:'<,'>:w !python3<CR>

autocmd FileType go set noexpandtab

autocmd FileType html :syntax sync fromstart

autocmd FileType html,javascript,css,json,yaml,sh

\ setlocal ts=2 sts=2 sw=2 expandtab

augroup END

Learning

- [A Minimal Vimrc](#)
- [Configure Vim with vimrc](#)

Cheat Sheets

PDF: [vim-commands-cheat-sheet-by-pnap.pdf](https://vim-command-commands-cheat-sheet-by-pnap.pdf)



Vim Cheat Sheet



Created by @dan_nanni on Instagram



Editing Mode

: command mode
ESC normal mode
i insert mode
v visual mode, highlight characters
V visual line mode, highlight lines

Cursor Movement

Character **h** (left) **j** (down) **k** (up) **l** (right)
Word **w** (next start) **b** (prev) **e** (next end)
Line **0** (start) **\$** (end) **^** (first non-blank)
Screen **Ctrl+f** (page down) **Ctrl+b** (page up)
File **gg** (first line) **G** (last line)

Editing Operation

Insert **i** (before cursor) **I** (line beginning)
a (after cursor) **A** (line end)
o (new line below) **O** (new line above)
Delete **x** (current char) **dd** (or **:d**) (current line)
D (to end of line) **dw** (word)

Copy and Paste

Copy **yy** (or **Y**) (current line) **3yy** (3 lines)
yiw (current word without whitespace)
yaw (current word with whitespace)
y\$ (to end of line)
Paste **p** (paste after) **P** (paste before)

Undo and Redo

u Undo the last action
U Undo all latest changes made to the line
Ctrl+r Redo the last action

Search and Replace

Search **/pattern** (from current cursor)
n (next match) **N** (prev match)
Replace **:s/old/new/g** (replace in cur line)
:%s/old/new/g (replace in all lines)
:%s/old/new/gc (replace with confirm)

File Operation

:w Save current changes
:w <filename> Save changes to another file
:q Exit but it will fail with unsaved change
:q! Exit without saving changes
:wq (or **:x**) Save changes and exit
:e <filename> Open a new file

Window Management

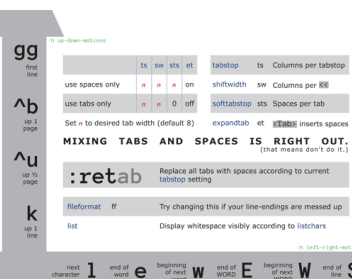
:split (or **:sp**) split window horizontally
:vsplit (or **:vs**) split window vertically
:new open a new window horizontally
:vnew open a new window horizontally
Ctrl-w switch cursor between windows
Ctrl-w r rotate window positions

Shell Command

:shell Open separate command prompt
exit Exit command prompt

Customization

:set number Show line numbers
:set tabstop=<N> Change tab width
:set hlsearch Highlight search matches
:set autoindent Enable auto indentation



j down 1 line	p paste after cursor	P paste before cursor	^ return to Normal mode
u undo	Ar redo	. repeat	
gd down 2 page	gf find file under cursor in parent directory	dd delete current line	yy yank current line
x delete character after cursor	% jump to matching pair	i insert	r replace char under cursor
af down 1 page	X jump to line 1	o jump back	I jump forward
	zz center screen on cursor	zt align top of screen with cursor	zb align bottom of screen with cursor
G	= auto-indent current line	<< shift current line left by <code>shiftwidth</code>	>> shift current line right by <code>shiftwidth</code>

Using **^_** to return to Normal mode lets you keep your fingers on the home

COOL INSERT MODE STUFF

$\wedge W$	delete word before cursor	$\wedge U$	delete line before cursor
$\wedge R r$	insert the contents of register r	$\wedge R =$	use the expression register (try <code>find=10</code>)

Diagram illustrating the effect of shiftwidth on the time interval between two events. The horizontal axis represents time. The distance between two points is labeled 'shiftwidth'. The time interval between the two points is labeled 't'.

COMMAND-LINE MODE ONLY

Put `moreapp $(C-8)$(and('X','X'))/.*'(C8)` in your `vimrc` so you can type `xx` in Command-line mode to refer to the directory of the current file, regardless of `pwd`.

Supply `xx` as a range to the `:%substitute` command to run it on every line in the file.

```
:%s/Scribbl/Design/      "Scribbled" -> "Designed"
```

Specify the "g" flag to apply the substitution to every match on a line.

```
s:/[dla]//g              "badly" -> "by"           :h s_flags, :h /[]
```

Minic supports many regular expression features:

```
:s/.k/ax/      "Mook" -> "Max"      :h wst.27. :h /
```

Use `.` instead of `/` if you want to search across multiple lines.

```
:%s/heat\..+Bungler/anto/      "Cheatsheet\nBungler" -> "Cantor"      :h /
```

```
Special escapes can be used to change the case of substitutions.  
:s_\(f, \)_\U\1E_ "foobar" => "FOObar" :h sub-replace-special  
Use :global to perform a command on matching lines.
```

```
:g/foobar\delete Delete all lines containing "foobar"
```

```
:s_\d_\s=smatch(0) + 1_g      "10 25" -> "21 36"      # sub-replace-1
```

<code>:h <i>cmd</i></code>	Normal mode <i>cmd</i> help
<code>:h <i>i_cmd</i></code>	Insert mode <i>cmd</i> help
<code>:h <i>v_cmd</i></code>	Visual mode <i>cmd</i> help
<code>:h <i>c_cmd</i></code>	Command-line editing <i>cmd</i> help
<code>:h <i>:cmd</i></code>	Command-line <i>cmd</i> help
<code>:h <i>'option'</i></code>	<i>Option</i> help
<code>:helpgrep</code>	Search through all help docs!



et.com

hidden	hid	Lets you switch buffers without saving	Use g instead of h when beginning text object to include delimiters or surrounding whitespace. For example, d[g] will change "(foo)" into "()", but d[h] will delete the parentheses as well.
laststatus	ls	Show status line never (0), always (2) or with +2 windows (1)	
research	hls	Highlight search matches. Also see 'highlight'	
number	nu	Show line numbers	
showcmd	sc	Show commands as you type them	
ruler	ru	Show line and column number of the cursor	
backspace	bs	Set to "2" to make backspace work like same editors	
wrap		Control line wrapping	
background	bg	Set to 'dark' if you have a dark color scheme	

REGISTERS vs CLIPBOARDS

All commands that delete, copy, or paste text use registers. To change which register is used by a command, type the register before the command. The default register is called the "main register," and it is invoked by a pair of double-quotes (""). Typing "" or "" is the same as typing "register". Think of the first "" as a short way of saying "register", so "" is pronounced "register" and "" is "register".

registers

Use this to
view all current
register key
mappings. Read
each key
for a guide
on which keys
are best for your
own custom
mappings. Get
used to my
help system -
it's a fantastic
resource!

Notepad
File Edit Format View Help
Registers
Registers are used to store text that is copied or pasted. The default register is the main register, which is accessed by pressing Ctrl+V or Ctrl+C. Other registers are accessed by pressing Ctrl+V or Ctrl+C followed by a number from 1 to 9. The registers are: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100.

"2"	19	register	Every time "19" is written to, its contents is pushed to "20" , and so on
"big"		big delete	
"_"		Small delete	Contains the last text we deleted within a single line
"+"		System clipboard	If we CG register code smile upon "x" , this register reads and writes to your system clipboard.
"a"-"z"		Named registers	26 registers for you to play with
"A"-"Z"		Append registers	Using upper-case to refer to a register will append to it rather than overwrite it
"q"		Record	Record into contents of register "x" . Stop recording by hitting q again
"@r"		Playback	Execute the contents of register "r"
"@@"		Repeat last playback	Repeat the last @r , this is particularly useful with a count

Modes & Controls

Command Mode ESC (commands preceded by :)

Insertion Mode Entered on insertion or change

Starting VI (command line)

vi <filename>	Edit <i>filename</i>
vi -r <filename>	Edit last version of <i>filename</i> after crash
vi + n <filename>	Edit <i>filename</i> at line <i>n</i>
vi + <filename>	Edit <i>filename</i> at end of file
vi +/str <filename>	Edit <i>filename</i> at first occurrence of <i>str</i>

In insertion mode the following should be preceded by ESC:

:w	Save
:x	Save & Exit
:q	Exit if no changes made
:q!	Exit & discard any changes

Cursor Navigation

h or ◀	Cursor left
j or ▼	Cursor down
k or ▲	Cursor up
l or ▶	Cursor right
w	Next word
W	Next blank delimited word
b	Start of word
B	Start of blank delimited word
e	End of word
E	End of blank delimited word
(Back a sentence
)	Forward a sentence
{	Back a paragraph
}	Forward a paragraph
0	Beginning of line
\$	End of the line
1G	Start of file
G	End of file
:n	<i>n</i> th line of file
f<char>	Forward to <i>char</i>
F<char>	Back to <i>char</i>
H	Top of screen
M	Middle of screen
L	Bottom of screen
%	Matching bracket
gg	Start of document

Inserting Text

i	Insert before cursor
a	Append after cursor
I	Insert before line
A	Append after line
o	Add new line after current line
O	Add new line before current line
r	Overwrite one character
R	Overwrite many characters
:r <file>	Reads <i>file</i> and inserts it after this line
p	Put after the position or line
P	Put before the position or line
C	Rewrite the whole line

Deleting Text

x	Delete character to right of cursor
X	Delete character to left of cursor
D	Delete the rest of line
dd or :d	Delete current line
ndw	Deletes the next <i>n</i> words
ndb	Deletes the previous <i>n</i> words
ndd	Deletes <i>n</i> lines starting with current
:x,yd	Delete lines <i>x</i> through <i>y</i>
:r <file>	Reads <i>file</i> and inserts it after this line
d{nav_cmd}	Overwrite many characters
:r <file>	Reads <i>file</i> and inserts it after this line

Searching

/string	Search forward for string
?string	Search backwards for string
n	Go to next match
N	Go to previous match
:set ic	Ignore case while searching
:set noic	Case-sensitive searching
:set nu	Turn on line numbers
:x,yg/str	Search for <i>str</i> from line <i>x</i> to line <i>y</i>
:g/str/cmd	Run <i>cmd</i> on lines containing <i>str</i>
*	Search for next instance of current word
#	Search for last instance of current word

Replacing

:s/pt/str/flag Replace pattern with string

Flags

g	Replace all occurrences of pattern
c	Confirm replaces
&	Repeat last :s command

Other

u	Undo last change
J	Join lines
nJ	Join next <i>n</i> lines
.	Repeat last command
U	Undo all changes to line
:N	Open split screen
v	Visual mode
ctrl + c	Escape insert mode



