Vim——程序猿最爱的文本编辑工具

——*By Rukawa Y.*

什么是Vim

Vim *(Vi IMproved)* 是一个高度可配置的文本编辑器，起源于一个名为Vi的更早期的文本编辑器。Vim在Unix和Unix-like系统中非常流行，允许用户在不离开键盘的情况下完成大部分代码编辑任务，因此被许多程序员和系统管理员所喜爱。它的学习曲线相对较陡，但一旦熟练掌握，它将成为一个强大的文本编辑工具。

如何安装Vim

Linux（以Debian/Ubuntu为例）

Linux通常自带Vim。对于大多数Linux发行版，可以使用包管理器安装Vim：

```
sudo apt-get update
sudo apt-get install vim
```

MacOS

使用Homebrew包管理器安装Vim：

```
brew install vim
```

Windows

1. 使用Git安装

克隆Vim的Git仓库来安装Vim的最新开发版本：

```
git clone https://github.com/vim/vim.git
cd vim
make
sudo make install
```

2. 下载安装程序

- 访问Vim官方网站下载页面
- 下载适用于Windows的安装程序并运行，完成安装

> 如果想在Powershell中直接使用Vim命令，需要配置环境变量将Vim添加到Path。

安装完成后，在终端中输入Vim即可启动它。

Vim的三种模式

普通模式（Normal Mode）

命令模式是进入Vim后的默认模式。在这个模式下，可以使用键盘快捷键来移动光标、复制、粘贴、删除文本等，但不能直接编辑文本。此状态下敲击键盘动作会被 Vim 识别为命令，而非输入字符，比如此时按下`i`，并不会输入一个字符，`i`被当作了一个命令。

命令模式下的一些常用命令包括：

- `h`、`j`、`k`、`l`：分别用于左、下、上、右移动光标。
- `w`、`b`：分别跳到下一个、上一个单词的开头。
- `$`：移动到当前行的末尾。
- `0` 或 `Home`：移动到当前行的开头。
- `gg`：移动到文件的第一行。
- `G` 或 `Shift+G`：移动到文件的最后一行。
- `dd`：删除当前行。
- `yy`：复制当前行。
- `p`：粘贴复制的文本。
- `/`：按下后下方出现查找框，输入需要查找的字符，按下回车查看搜索结果
- `*`：查找当前光标下的单词

此外，在输入一个命令前加上数字n的话，该命令就会重复n次。

插入模式（Insert Mode）

要进入插入模式，可以在命令模式下按 `i`、`I`、`a`、`A`、`o` 或 `O` 等键。在这个模式下，你可以像在其他文本编辑器中一样直接输入文本。

- 插入模式下的命令与普通模式不同，主要是文本输入命令。
- 要退出插入模式并返回普通模式，可以按 `Esc` 键。

末行模式（Command Mode）

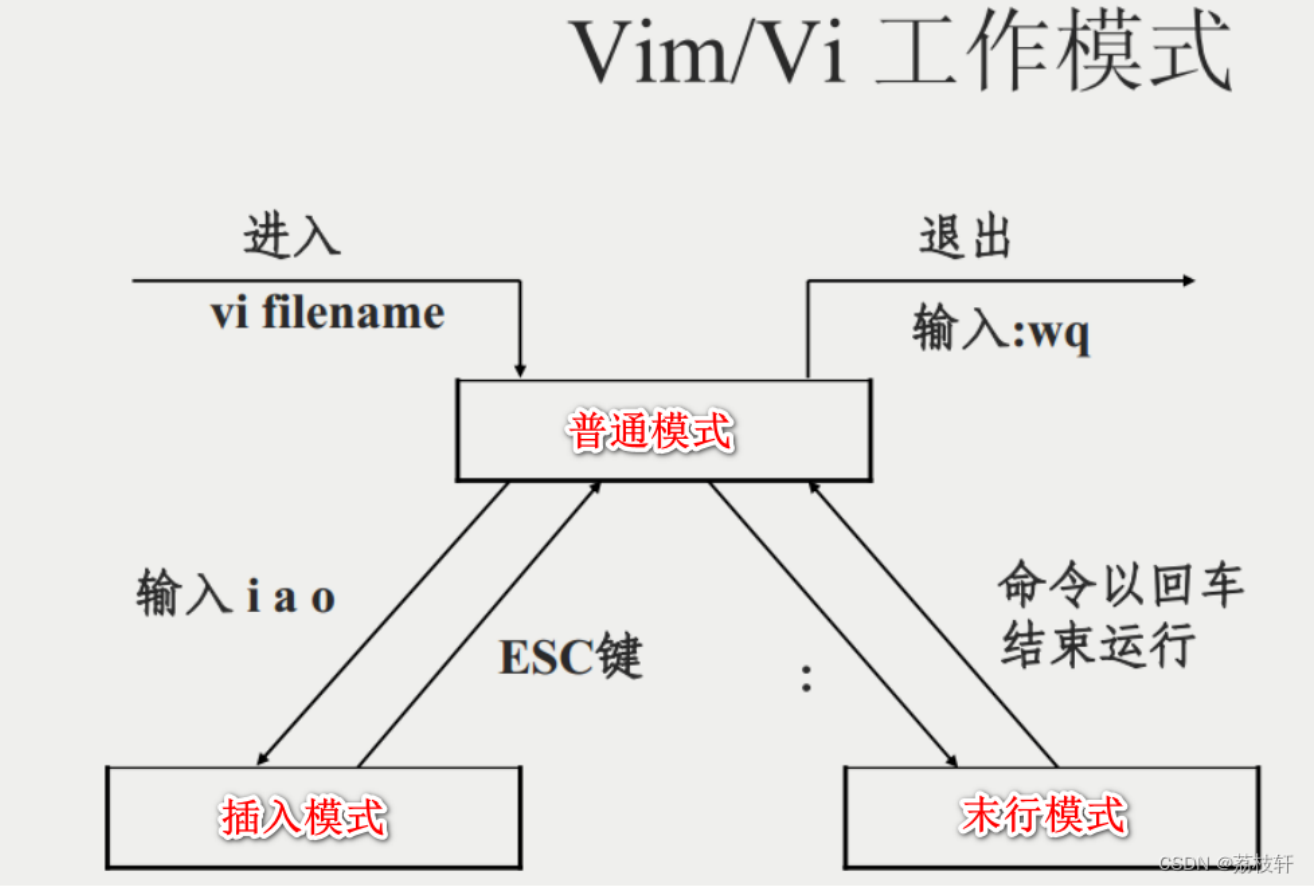
末行模式用于执行对文件的命令，如保存文件、退出 Vim、查找和替换文本等。要进入命令模式，在普通模式下输入一个冒号 `:`即可。

命令模式下的常用命令包括：

- `:w`：保存文件。
- `:q`：退出 Vim。
- `:wq` 或 `:x`：保存文件并退出 Vim。
- `:q!`：不保存更改并退出 Vim。
- `help/:h`：查看Vim在线帮助文档
- `:find filename`：在当前目录下查找文件。
- `:%s/old/new/g`：查找并替换文本（`%` 表示对整个文件进行操作）。

执行完命令后，Vim 会自动返回普通模式。

三种模式的关系如图所示：



Vim进阶操作

分屏操作

在Vim中，分屏允许你在同一窗口中查看和编辑多个文件，这对于比较文件或同时编辑多个文件非常有用。

- 水平分屏

  在普通模式下，输入`:split`或`sp`，可以将当前窗口水平分割成两个窗口。

  ```
  :split 或 :sp
  ```

  可以在命令后面添加文件名来在新窗口中打开该文件，例如 `:split filename`。

- 垂直分屏

  在普通模式下，输入 `:vsplit` 或 `:vsp`，可以将当前窗口垂直分割成两个窗口。

  ```
  :vsplit 或 :vsp
  ```

同样，可以在命令后面添加文件名来在新窗口中打开该文件，例如 `:vsplit filename`。

查找替换

查找和替换是 Vim 中非常强大的文本处理功能，允许你快速定位和修改文本。

- **查找**： 在普通模式下，输入 `/keyword` 可以向下查找“keyword”，输入 `?keyword` 可以向上查找“keyword”。

  ```
  /keyword 或 ?keyword
  ```

- **替换**： 在命令模式下，输入 `:s` 可以替换文本。以下是一些替换的例子：
  - 替换当前行的第一个匹配项：

    ```
    :s/old/new
    ```

- 替换当前行的所有匹配项：

```
:s/old/new/g
```

- 替换文件中从第10行开始的所有匹配项：

```
:10,20s/old/new/g
```

- 替换整个文件中的所有匹配项：

```
:%s/old/new/g
```

宏

宏允许录制一系列操作，然后重复执行这些操作，这对于自动化重复任务非常有用。

- **录制宏**： 在普通模式下，选择一个寄存器（如 a 到 z），然后输入 q 开始录制宏。

```
qa
```

  执行你想要录制的操作，然后按 q 结束录制。
- **执行宏**： 在普通模式下，输入 @ 加上你录制宏时使用的寄存器字母来执行宏。

```
@a
```

  这将重复你之前录制的所有操作。
- **多次执行宏**： 你可以在 @ 后面加上数字来多次执行宏，或者使用 : 命令。

```
@3a 或 :3@a
```

  这将执行宏三次。

宏是提高编辑效率的强大工具，一旦你掌握了它们，就能够快速完成复杂的编辑任务。

写在最后

初学Vim的大家肯定会觉得Vim的操作太复杂，太难记忆，很容易就被劝退。但是Vim的学习曲线虽然陡峭，一旦花时间精通之后，带来的效率增长是指数级的。“非常地难学，但是却不可思议地好用”，用来形容Vim非常合适。

> Q: 觉得命令太多太难记了怎么办?

没关系，我们有**CHEAT SHEET**!

以下是一张汇总了绝大多数Vim操作的键位图，供大家在懵的时候参考：

Esc Normal | *Revision 2.0 Sept. 11, 2011* | Vim 7.3+ `:version`

Vim Cheat Sheet for Programmers

Copyleft © 2011
May be freely distributed! Sharing is Caring.
http://michael.PeopleOfHonorOnly.com/vim/

HOW-TO** make Vim not suck Out of the Box:* ***:help statusline :set nocompatible ruler laststatus=2 showcmd showmode number ***Search*** :set incsearch ignorecase smartcase hlsearch ***Remove useless splash screen*** :set shortmess+=I

Best tips: http://vim.wikia.com/ **Best scripts:** http://www.vim.org/scripts/index.php :map <F9> :e $HOME/_vimrc<CR> :map <F6> :so $HOME/_vimrc<CR>

Number row

- Ctrl ` — **~** toggle case / **`•** goto mark [13]
- Ctrl 1 — **!** extern filter / **1** [1]
- Ctrl @ — **@•** play macro / **2**
- Ctrl 3 — **#** prev identifier / **3** [3]
- Ctrl 4 — **$** →| / **4**
- Ctrl 5 — **%** goto match / **5**
- Ctrl ^ — **^** soft |← / **6**
- Ctrl 7 — **&** repeat :s / **7**
- Ctrl 8 — ***** next identifier / **8** [8]
- Ctrl 9 — **(** begin sentence / **9**
- Ctrl 0 — **)** end sentence / **0** hard |←
- Ctrl _ — **_** cur line / **-** ↑
- Ctrl = — **+** ↓ / **=** auto-format [2]

Q row (Tab)

- [14] block select — **Q** ex mode / **q•** record macro
- ^w• window… — **W** WORD ↘ / **w** word ↘
- scroll line ↑ — **E** end WORD ↘ / **e** end word ↘
- [12] :redo — **R** Replace / **r** replace char
- [10] ctags return — **T•** ← until char / **t•** until char →
- scroll line ↓ — **Y** copy line / **y** copy [0 2]
- half page ↑ — **U** undo line / **u** undo
- Ctrl I — **I** insert |← / **i** ⤶ insert
- prev mark — **O** open ↑ / **o** open ↓
- [9] ↑ — **P** paste ↑ / **p** paste ↓ [0]
- Normal — **{** ↖ paragraph / **[•** misc.
- ctags identifier — **}** paragraph ↘ / **]•** misc.

A row (Caps)

- [7] incr. # — **A** append →| / **a** append ⤴
- **S** subst line / **s** subst char
- [10] half page ↓ — **D** del →| / **d** del [0 2]
- page ↓ — **F•** ← find char / **f•** find char →
- file/cursor info — **G** goto eof / goto line# / **g•** extra [6]
- Ctrl H — **H** Top screen / **h** ←
- [15] Ctrl J — **J** Join lines / **j** ↓
- **K** man page identifier / **k** ↑
- redraw — **L** Bottom screen / **l** →
- Ctrl ; — **:** cmd line / **;** "next" f/F/t/T [16]
- Ctrl ' — **"•** register [0] / **'•** goto mark |←
- Ctrl \ — **|** |← goto col# / **\•**

Z row (Ctrl ^ / Shift ⇧)

- :suspend — **Z•** quit [4] / **z•** extra [5]
- [7,11] decr. # — **X** ← del char / **x** del char →
- Normal / Cancel — **C** change →| / **c** change [0 2]
- block select — **V** select lines / **v** select chars
- page ↑ — **B** ↖ WORD / **b** ↖ word
- [9,16] ↓ — **N** "prev" find / **n** find "next"
- [15] Ctrl M — **M** Middle screen / **m•** set mark
- Ctrl , — **<** undent [2] / **,** "prev" f/F/t/T [16]
- Ctrl . — **>** indent [2] / **.** repeat cmd
- Ctrl / — **?•** find ↖ / **/•** find ↘

Unused & Duplicate keys
`\ Ctrl-K Ctrl-S (free)`
`Ctrl-L (redraw)`
13 `` ` near dup of ' ``
14 `Ctrl-Q = Ctrl-V`
15 `Ctrl-J = Ctrl-M = ^N`

16 The search direction is relative; **next** is the initial direction, **previous** is the opposite direction. `n ;` repeat same initial direction find. `N ,` repeat opposite initial direction find. **Note**: `; ,` only searches cursor line, `n N` searches buffer.

Legend:

Macro	Register name (0-9a-zA-Z) required
Op	Motion req.; act between cursor & dst
Cmd	Command
Ins	Command and enter insert mode
Move	Moves cursor or defines range for op
Find	Search (↖ = reverse, ↘ = forward)
tag	ctags / diffs / folding
Code	Code formatting, whitespace, etc.
Extra	Extended functionality; req. extra chars
•	Char arg req. `g z Z ^w ' " ` …`

Modes

`:help modes`

n	Normal	`Esc ^[ ^c`
i	Insert	`a i r s`
v	Visual	`v V ^v ^q`
o	Op pending	`c d y < >`
c	Command Line	`: / ? !`

word: `Foo` `(` `src` `,` ␢ `dst` `,` ␢ `len` `);`
Note: There is no whitespace in-between 'Foo(src,' but before/after 'dst,'
WORD: `Foo ( src ,` ␢ `dst ,` ␢ `len );`

Startup

`vim <filename> +123`	**goto line 123**
`vim <file> … -t Foo`	**edit at tag 'Foo'**
`vim <file> … -c "/Foo"`	**cmd: find 'Foo' & edit**
GUI `vim -g` or `gvim`	start GUI ver.
GUI Linux	`:set guifont=ProggyTinyTT\ 12`
GUI OSX	`:set guifont=ProggyTiny\:h11`
diff	`gvimdiff <file1> <file2> [<file3>]`

bug **Broken Keys** ~~Ctrl-I = Tab, Ctrl-[= ESC~~
Vim is *still* **unable** to map certain keys for your own use…
§ **Caps, Ctrl-1, Ctrl-Shift-1, Ctrl-I, Ctrl-\, etc.**

0 *See:* `src/ops.c -c "/valid_yank_reg"` *for "• reg. names*
6 *See:* `src/normal.c -c "/nv_cmds"` *for g• extra cmds*
11 *See:* `src/edit.c -c "/ctrl_x_msgs"` *for ^x• insert cmds*

:help cmdline

`:r file`	insert file		
`:w`	save	`:gui`	switch to GUI
`:q`	quit	`:q!`	quit w/o save
`:e <file>`	edit file in new buffer		
`:source %`	**exec cmds in cur file**		
`:exec '...'`	**do cmd**		

:help movement

soft `^`	**|← Start of Line**	1st non-whitespace
hard `0`	**|← Start of Line**	column 0
`$`	**→| End of Line**	
`|` move col 0	`#|` **move col #**	
`^b` page ↑	`^f` page ↓	
`^u` ½ page ↑	`^d` ½ page ↓	
`^e` scroll line ↑	`^y` scroll line ↓	
`1g` start of file	`0g` end of file	
`#g` **goto line #**	`G` **end of file**	
`[[` **begin this func {**		
`]]` **begin next func {**		

:set matchpairs=(:),{:},[:],<:>,?:\:
`%` **goto matching { } < > [ ]**

:help range

`:s/Foo/Bar`	find Foo replace w/ Bar
`:s/Foo/Bar/g`	...all instances on line
`:%s/Foo/Bar`	**apply to whole file**
`.,.+#`	**cur line, cur line + # lines**
`$` **last line**	`'<` start of select
	`'>` end of select

Tags

`:ts`	list active tags
`^]`	**jump to tag under cursor**
`^t`	**restore cursor before tag jump**
`^p`	**complete word**
`:ta Foo`	manual jump to tag 'Foo'

Diff

`[c` **prev diff**	`:hi DiffAdd guifg=#rrggbb`
`]c` **next diff**	`:hi DiffChange guibg=#rrggbb`
`:diffupdate`	`:hi DiffText gui=none`
resync	`:hi DiffDelete`

Folding

`zR`	**fold remove**
`zo`	fold open
`zc`	fold close
`zi`	**invert all**
`zr`	fold reduce
`zm`	fold more

Changes

`:changes`
`g;` older change
`g,` newer change

Syntax

`:syntax enable`
`:set filetype=`
c cpp sh make perl python
Note: chose only ONE type!

:help recording

`q•`	start recording
`@•`	playback
`q`	stop recording
`@@`	repeat

Convert <eol>

`:set fileformat=`
unix or *dos* or *mac*
then `:w` to convert

Code

`= < > << >>`

`:set backspace=indent,eol,start`	**allow backspace join lines**
`:set shiftwidth=#`	**indent width for ai**
`:set autoindent!`	**toggle auto-indent**
`:set lisp`	**lisp indent mode**
`:set tabstop=#`	**set tab stop every #th col**
`:set expandtab!`	**toggle hard/soft tabs**
`:set listchars=…`	*tab:>-:trail:-,nbsp:%,eol:$*
`:set list!`	**toggle whitespace**
`:set colorcolumn=80`	**visible right margin indicator**
`noremap + :s/^/\/\/<CR>`	**block comment**
`:set numbers!`	**toggle line numbers**
`:set wrap!`	**toggle linewrap display**
`:set showmatch`	**highlite matching ()**
`noremap - :s/^\/\///<CR>`	**uncomment**

Commands

`\ :map \ :Explore<CR>` manually type <,C,R,>

§0	`"•`	before del/copy/paste to use register
	`"+x`	cut to system clipboard reg. '+'
	`"+gP`	paste from system clipboard
1		Number before any action repeats it
	`2p`	paste twice — `3.` repeat thrice
2		Repeat op to act on current line
	`yy`	copy line — `dd` del line
	`<<`	**undent line** — `>>` **indent line**
3	`#`	highlight words under cursor
4	`ZZ`	save & quit — `ZQ` quit w/o save
5	`zz`	**center cursor line in window**
	`zh`	**scroll left** — `zl` **scroll right**
	`zt`	scroll top — `zb` scroll bottom
§6	`gg`	top of file
	`gf`	**open file under cursor**
7	`^a`	incr # under cursor (Dec / Hex)
	`^x`	decr # under cursor (Dec / Hex)
8	`*`	start a "new" search

Insert mode

9	`^p`	**prev auto-complete** — `^n` **next**
10	`^d`	**undent** — `^t` **indent**
§11	`^x• ^f`	filename completion
	`^s`	spelling — `:set spell!`
	`^k`	dictionary — `]s` **next bad**
	`^t`	thesaurus — `:help spell`
12	`^r•`	paste register 0-9a-zA-Z or …
	`+`	**clipboard** (or '*') — `:help c_CTRL-R`
	`"`	last del/copy — `%` filename

:buffer

`:buffers`	list files
`:new`	blank file/buffer
`:bn`	**next file**
`:bp`	prev file
`:bd`	**close file**
`:bd!`	force close
`:set lines=#`	
`:set columns=#`	
`:winpos # #`	GUI

Windows

`:help windows`
`^w•` or `:wincmd •`

`w`	**:switch to next**
`c`	**:close!**
`n`	:new
`s`	**:split** horz.
`v`	:vsplit vertical
`o`	:only maximize
`=`	**all same size**
`h`	move to win ←
`j`	move to win ↓
`k`	move to win ↑
`l`	move to win →

`:sp [<filename>]` *edit in split window*

Cursor Bookmarks

`:marks` `` ` ' ^o ``

`ma`	mark local 'a'
`'A`	goto global 'A'
`''`	**prev location**

File / Directory

`:Explore` or `:e .`
`:set browsedir=…`
one of *buffer last*