Cheatography

Juanjux's Vim Cheatsheet Cheat Sheet

by Juanjo Álvarez (juanjux) via cheatography.com/19291/cs/2344/

Cursor movement and text motions

h/j/k/l	motion left/up/down/right
f / **F** *char*	motion to char in current line
t / **T** *char*	motion 1 before/after char in current line
w / **B**	motion to start of next / prev word
e / **ge**	motion to end of next / prev word
/**[regexp]**	search
0	motion to start of line
^ / **ç**	motion to start of line text
$	motion to end of line
[number] **G** / **gg**	go to line at [number]
gg	motion to start of buffer
G	motion to end of buffer
o (visual mode)	move cursor to start/end of selection
(/)	start / end of phrase
{ / }	start / end of paragraph
[{/]}/[(/])	motion to next / prev { }( )
<c-o> / **<c-i>**	navigate cursor location history
:goto *[number]*	go to byte at position *number* in buffer
* / #	search next / prev occurrence of word under cursor
<ldr>**ta**	jump to tag *user*
<c-j> / **<c-k>**	pagedown / pageup *user*

Cursor movement and text motions (cont)

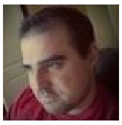
<ldr>**e**	shortcut jump to word *user|plugin:easymotion*
<ldr> <ldr>**s** *[char]*	shortcut jump to *char* bidirectional *plugin:easymotion*
<ldr> <ldr>**e**	shortcut jump to end of word forward *plugin:easymotion*
<ldr> <ldr>**ge**	shortcut jump to end of word backward *plugin:easymotion*

Yank/Paste

register: "*	system clipboard (Windows)
register: "+	system clipboard (Linux)
register: "%	buffer name
register: *=	output of system()
:reg	list of registers
[register]**y**[motion]	yank (yy for current line)
p / **P**	paste at the / before current position
:pu / <ldr>**,**	paste below current line *user*
gp	paste and move the cursor to the end
<ldr>**yy**	show yankring *user|plugin:yankring*
<c-p> *(after pasting)*	cycle between recent yanks *plugin:yankring*
<ldr>V	select last pasted test *user*
<c-rp> *register*	yank from register in insert mode

My Plugins

Pathogen	to manage plugins
Yankring	yank between different instances of Vim. Cycle when pasting
Project	simple project management. \C to create a new project, \R to update files
NERD_Commenter	easily comment/uncomment code
Matchparen	shows matching delimiter
Csapprox	use GVim colorschemes in console Vim
Airline	awesome status bar
Tagbar	jumpable tag list
Vimwiki	personal Wiki
Syntastic	Validate syntax for several languages
EasyMotion	jump to any word/line/character with few keystrokes
Unite	Fuzzy search for buffers/files/etc
Matchit	improves VIm's % command to understant more delimiters
MatchTagAlways	highlight matching HTML tags
Tabular	align things
Emmet	Quickly edit HTML and move between tags
YouCompleteMe	excellent completion
VimColorSchemes	lot of colorschemes
Surround	operations on delimiters
Jdaddy	add a JSON text object



My Plugins (cont)

GTFO	shortcuts to open GUI file explorer or terminal in buffer CWD
Ack	:Ack search (but use ag instead of Ack!)
Repeat	the repeat operator (dot) will be able to repeat some plugin actions
LustyJuggler	Show buffers as a list, switch with a number
Rename	:Rename current buffer and the associated file

Operators: [repeat]operator[motion]

d	delete, dd for line
c	change (delete and enter insert), cc for line
y	yank (yy for line)
gu / **gU**	to lowercase / uppercase
!	filter through external program
=	auto indent
gw	justify
> / **<**	indent / dedent

[repeat] [operator] [motion or text object]
Examples:
10dj (delete 10 lines down)
v3w (select 3 words ahead)
yiw (yank the inner-word object)
gwip (justify the inner-paragraph object)
3diw (delete the next 3 inner-words)

Text edition (see also operators & motions)

i	enter insert mode
a	enter insert mode after cursor position (append)

Text edition (see also operators & motions) (cont)

x	delete char under cursor
r*[newchar]*	replace char under cursor
I	move to start of line and insert
A	move to end of line and append
D	delete from cursor to end of line
C	change from cursor to end of line
o / **O**	create new line and insert below / above cursor position
<ldr>**o** / <ldr>**O**	create new line above / below without going to insert mode
:tab /*[char]*/l1	tabularize from *[char]* left align 1 space *plugin:tabularize*
ys*[motion] [delimiter]*	surround motion/object with delimiter *plugin:surround*
S[visual selection] *[delimiter]*	surround visual selection with delimiter
cs*[delimiter] [newdelim]*	change surrounding delimiter for *newdelim* *plugin:surround*
ds*[delimiter]*	delete surrounding *delimiter* *plugin:surround*
<ldr> **c<space>**	comment / uncomment
J	join lines

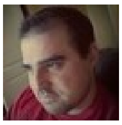
Text edition (see also operators & motions) (cont)

[operator] <ldr>**e**	shortcut operate by lines with *plugin:easymotion*
<c-w> (insert mode)	delete one word back

Ex Commandline

range: **%**	full buffer
range: **.** *(dot)*	current line
range: **$**	last line
range: +-N	+-N lines from current position
selector: **g**	global command (selects lines matching regexp)
selector: **v**	inverse (select lines NOT matching regexp)
operator: **d**	delete
operator: **!***[program]*	replaces with output from external program
operator: **s**/*old*/*new*	replaces *old* with *new*
operator: **y**	yank
operator: **normal** *[commands]*	run *commands* over every line in the range or selection
operator: **put** *[register]*	put the register content below the current line

The format is one of:
:[range][selector]/[regexp]/[operation]
:[range][operation]
:[operation]
Example:
:20,30g/FIXME/d => delete lines from 20 to 30 with the string FIXME
:.,$v/DONTDELETE/d => delete lines without the string DONTDELETE from current position to end of buffer.
:%s/old/new => replace all instances of old with new
:.!ls => insert output of ls replacing current line



Emmet plugin for HTML

<c-y> *(insert mode)*	expand HTML abbreviature like: *Div#page>div.logo+ul>lБ*>a*
<c-y> **,** *(visual mode)*	wrap selection inside tag (asks for tag)
<c-y> **n**/**N**	jump to next / prev editable point
<c-y> **k**	remove current tag
<c-y> /	comment tag
<c-y> **a**	converts URL to <A> tag
<c-y> **A**	converts URL to <A> tag autoloading site description
<c-y> **c**	prettify code with CSS classes by type
<c-y> **d**	sleect tag surrounding the cursor
<c-y> **D**	select next inside tag or next tag
<c-y> **i**	update image size
<c-y> **j**	join separate block: <p></p> ⇒ <p/>
<c-y> **,**	with "lorem" in insert mode, writes lorem ipsum

Spellcheck

<ldr>**ss**	spanish spell check ^{user}
<ldr>**se**	english spell check
<ldr>**sn**	disable spell check
z=	suggestions for word under cursor
]s / **[s**	next / prev misspelled word
zg	add word to user dictionary

Text Objects i/a[object]

l	char
w/**W**	word / WORD
s	sentence
(/**)** or **b**	(block)
{/**}** or **B**	{block}
j	JSON block, use gw to prettify *plugin:jdaddy*
p	paragraph

Windows / Buffers / Tabs / Splits

:w	save buffer
:q/**:x**	exit / save and exit
:e *[path]*	open file at *path*, use "." for netrw browser
:bd	delete buffer (doesn't delete file)
<ldr>**v**	create vertical split and jump to it ^{user}
<ldr>**tn**	new tab ^{user}
<c-h> / **<c-l>**	move to left / right tab ^{user}
*[number]***gt**	go to tab *[number]*
:sp	create horizontal split
<ldr>**cv**	close current split ^{user}
<ldr>**bt**	open all buffers in tabs ^{user}
:tabm *[pos]*	move tab to pos (can be +/- pos)
<space><space>	Unite buffers/MRU *user|plugin:unite*
<space>**b**	Unite buffers $^{plugin:unite}$

Windows / Buffers / Tabs / Splits (cont)

<space> **m**	Unite most recently used *plugin:unite*
<space> **f**	Unite search files recursively *plugin:unite*
<c-w>**T**	Convert split to tab
<c-w>**r**	Rotate split positions
:tabo	close all tabs except current
:w!!	save buffer as root with sudo ^{user}

Marks

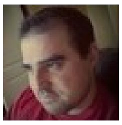
m*[mark]*	create mark *[mark]*
'*[mark]*	jump to start of line with *[mark]*
` *[mark]* / **¡**	jump to *[mark]*
mark: **0** *(zero)*	last file edited
mark: **'**	before last jump
mark: **,**	last edition point

Undo / Redo

u	undo
<c-r>	redo
:earlier *[time spec]*	s:second / m:minute / h:hour / d:days
:later *[time spec]*	s:second / m:minute / h:hour / d:days

Misc

<c-o> *(insert mode)*	run a command in insert mode
<ldr>**cdt**	update D tags ^{user}
<ldr>**ct**	clear trailing whitespace *user*
<ldr>**sv**	reload .vimrc ^{user}



Misc (cont)

<ldr>**rr**	systax on and off (to reset colors) *user*
za / **zA**	toggle fold / class level
gof	open graphical file browser in buffer cwd *plugin:gtfo*
got	open command terminal in buffer cwd *plugin:gtfo*
<ldr>**<space>**	clear search result highlights *user*
<ldr>**gs**	save vim session *user*
<ldr>**css**	load vim session *user*
v / **V**	enter visual mode / by line
<c-v>	visual block selection (I to insert before, A to append after)
<ldr>**tb**	open tag bar split *user|plugin:tagbar*
<ldr>**P**	open project split *user|plugin:project*
<ldr>**sy**	syntactic check with syntactic *user|plugin:syntactic*
:Ack	search with Ack/ag
:windo diffthis	mark split for windiff (do twice)
. **(dot)**	repeat last change
:sort **[selection]**	sorts the selection

Misc (cont)

:Rename	rename current buffer and file *plugin:rename*
<ldr>**lj**	show buffer list, switch with a number *plugin:lustyjuggler*
@:	repeat last ex mode command
[Bash Shell] **set -o vi**	put bash commandline in vi mode
[Bash shell in vi mode] **<esc>v**	edit shell command in Vim

Tricks

xp in normal mode to exchange two chars

ddp to exchange two lines (or lower the current one one down)

In the Bash shell, use **set -o vi** to enable Vi mode. Then use <esc> and edit your command like in Vi or use **v** to edit your command line in Vim