Vim Tricks

Piotr Karamon

<2022-06-27 Mon>

Contents

1 Searching/quick jumping

- `* and #` search for a word under the cursor to search forward - * to search backward -#
- `gf` start editing the file whose name is under the cursor
- `gF` start editing the file whose name is under the cursor and jump to the line number following the filename
- `ge` go backwards to the end of the previous word
- '' jump to the last remembered line
- `two backticks` jump to the last remembered position
- `CTRL-O` Go to [count] Older cursor position in jump list (not a motion command).
- `<Tab> or CTRL-I` Go to [count] newer cursor position in jump list (not a motion command).
- `gD` go to the definition of word under the cursor in current file

- `gd` go to the definition of word under the cursor in current function

- `/regex/{n}` Makes the motion go to the nth line below the match, or above if n is negative. It also has the side effect of making the motion line-wise.

- `:g/regex/ex` Runs the Ex command only on the lines that match regex. So for example you can use g/regex/d to delete all the lines matching the regex. v is like g except it runs ex on all of the lines that don't match regex.

2 Vim surround

- `!` denotes the cursor position

- aliases: `"hello !world"`

- `b = ()`

- `B = {}`

- `a = <>`

- `r = []`

2.1 some examples

```
"hello !world"          ds"     hello world
[123+!4*56]/2           cs])    (123+456)/2
"Look ma, I'm !HTML!"   cs"<q>  <q>Look ma, I'm HTML!</q>
```

3 Miscellaneous

- `g8` print hex value of bytes used in UTF+8 character under the cursor

- `g<` dispay previous command output

- `g?` Rot13 encoding operator

- `g??` Rot13 encode current line

- `g?G` Rot13 encode to the end of file

- `gI` like I but always start at column 1

- `gR` enter Virtual Replace mode
- `gU{motion}` make motion uppercase the letters
- `gu{motion}` make motion lowercase the letters
- `gn` find the next match with the last used
- `gv` reselect the previous visual area
- `v_o` Goes to the other end of the visual block. Useful if you started one line too low or something. In block mode, it goes to the opposite diagonal corner: use v_O to go to the opposite horizontal corner.
- `z<CR>` redraw, cursor line to top of window, cursor on first non+blank
- `z-` redraw cursor line at the bottom of window
- `z.` redraw, cursor line to center of window, cursor on first non+blank
- `z=` give spelling suggestions
- `z...` commands about folds
- `<C-a> and <C-x>` increment and decrement a number
- `q:` Opens a history of your previous commands. You can operate on them like any Vim text, but changes aren't saved. But you can run a modified command with <CR>. This makes it really easy to quickly modify and rerun commands, or search for an old one to reuse.
- `q/, q?` Same as q: , except for searches.
- `gq` takes a logical line which is wrapped and splits it into smaller lines