

Vim tips

August 15, 2010

Contents

| | | |
|----------|--|-----------|
| 1 | Text manipulation | 4 |
| 1.1 | Searching | 4 |
| 1.1.1 | Search for declaration of subroutine/function under cursor | 4 |
| 1.1.2 | Search for visually highlighted text | 4 |
| 1.2 | Substitution | 5 |
| 1.2.1 | Filter all form elements into paste register | 6 |
| 1.2.2 | Substitute within substitution | 6 |
| 1.2.3 | Substituting a visual area | 7 |
| 1.3 | Global command display | 7 |
| 1.4 | Global combined with substitute (power editing) | 8 |
| 1.5 | Changing case | 8 |
| 1.6 | Reformatting text | 8 |
| 1.7 | Deletion without destroying buffer | 8 |
| 1.8 | Essential | 8 |
| 2 | File Manipulation | 10 |
| 2.1 | Exploring | 10 |
| 2.2 | Opening files & other tricks | 10 |
| 2.3 | Multiple files management | 10 |
| 2.4 | File-name manipulation | 11 |
| 2.5 | Command over multiple files | 11 |
| 2.6 | Sessions (set of files) | 11 |
| 2.7 | Modelines | 12 |
| 2.7.1 | Creating your own GUI Toolbar entry | 12 |
| 2.8 | Markers & moving about | 12 |
| 2.9 | Editing/moving within insert mode | 12 |
| 2.10 | Abbreviations & maps | 13 |
| 2.10.1 | Display RGB colour under the cursor eg #445588 | 13 |

| | |
|--|-----------|
| 3 Registers | 14 |
| 3.1 List your registers | 14 |
| 3.2 Appending to registers | 14 |
| 3.3 Using a register as a map (preload registers in .vimrc) | 14 |
| 3.4 Redirection & paste register | 14 |
| 3.4.1 Copy full path name | 15 |
| 3.5 Useful tricks | 15 |
| 4 Advanced | 16 |
| 4.1 Command line tricks | 16 |
| 4.2 External programs | 16 |
| 4.3 Recording | 16 |
| 4.3.1 Operating a Recording on a Visual BLOCK | 17 |
| 4.4 Quick jumping between splits | 17 |
| 4.5 Visual mode basics | 17 |
| 4.6 vimrc essentials | 17 |
| 4.6.1 Launching IE | 18 |
| 4.6.2 FTPing from vim | 18 |
| 4.6.3 Autocmd | 18 |
| 4.7 Conventional shifting and indenting | 18 |
| 4.8 Pulling objects onto command/search line | 18 |
| 4.9 Capturing output of current script | 19 |
| 4.10 Inserting DOS carriage returns | 19 |
| 4.11 Perform an action on a particular file or file type | 19 |
| 4.12 Inserting line number | 19 |
| 4.13 Numbering lines | 19 |
| 4.14 Advanced incrementing | 20 |
| 4.14.1 Create list starting from 223 incrementing by 5 between markers a,b | 20 |
| 4.14.2 Create a map for INC | 20 |
| 4.15 Digraphs (non alpha-numeric) | 20 |
| 4.16 Complex vim | 20 |
| 4.17 Syntax highlighting | 20 |
| 4.18 Preventions and security | 21 |
| 4.19 Taglist | 21 |
| 4.20 Folding | 21 |
| 4.21 Renaming files | 21 |
| 4.22 Reproducing lines | 21 |
| 4.23 Reading MS-Word documents | 22 |

| | |
|--|-----------------|
| <i>CONTENTS</i> | <i>CONTENTS</i> |
| 4.24 Random functions | 22 |
| 4.24.1 Save word under cursor to a file | 22 |
| 4.24.2 Delete duplicate lines | 22 |
| 4.24.3 Columnise a CSV file for display | 22 |
| 4.24.4 Highlight a particular csv column | 22 |
| 4.25 Miscallaenous commands | 22 |
| 4.26 Vim traps | 23 |
| 4.27 Help | 23 |
| 5 Fun | 24 |
| 6 Contact | 25 |
| 6.1 Author (David Rayner) | 25 |
| 6.2 Maintainer (Gavin Gilmour) | 25 |
| 6.3 Source | 25 |
| 6.4 Links | 25 |

1 Text manipulation

1.1 Searching

| Command | Description |
|---|--|
| /joe/e | cursor set to end of match |
| /joe/e+1 | cursor set to end of match plus 1 |
| /joe/s-2 | cursor set to start of match minus 2 |
| /^joe.*fred.*bill/ | normal |
| /^[A-J]\+/ | search for lines beginning with one or more A-J |
| /begin_. *end | search over possible multiple lines |
| /fred\s*joe/i | any whitespace including newline |
| /fred\ joe | search for fred or joe |
| /. *fred&.*joe | search for fred and joe in any order |
| /\<fred\>/i | search for fred but not alfred or frederick |
| /\<\d\d\d\d\> | search for exactly 4 digit numbers |
| /\D\d\d\d\d\> | search for exactly 4 digit numbers |
| /\<\d\{4\}\> | same thing |
| /\([^\0-9]\ ^\)\%. *% | search for absence of a digit or beginning of line |
| /^\n\{3} | find 3 empty lines |
| /\ (fred\).*\ (joe\).*\2.*\1 | using rexp memory in a search |
| /^\([^\,]*\)\{8} | using rexp memory in a search |
| /<\zs[^\>]*\ze> | search for tag contents, ignoring chevrons (:h /\zs) |
| /<\@<=[^\>]*>\@= | search for tag contents, ignoring chevrons |
| /<\@<=[^\>]*>\@= | search for tags across possible multiple lines |
| /i!-p{-}-i | search for multiple line comments |
| /fred\s*joe/i | any whitespace including newline (\s) |
| /bugs\(\.\)*bunny | bugs followed by bunny anywhere in file |
| :h _ | help |
| :bufdo /searchstr/ | multiple file search (use :rewind to recommence search) |
| :bufdo %s/searchstr/&/gic | multiple file search better but cheating (n, then a to stop) |
| ? http://www.vim.org/ | search backwards for a URL without backslashing |
| /\c\v([^\aeiou]&\a){4} | search for 4 consecutive consonants |

1.1.1 Search for declaration of subroutine/function under cursor

```
:nmap gx yiw/^(\(sub\<bar>function\)\s\+<C-R>"<CR>
```

1.1.2 Search for visually highlighted text

```
:vmap // y/<C-R>"<CR>
```

with spec chars

```
:vmap <silent> // y/<C-R>=escape(@, '\./.*$~[]')<CR><CR>
```


1.2 Substitution

```
:%s/fred/joe/igc
:%s/\r//g
:%s/\r\r/g
:%s= *$==
:%s= \+$==
:%s#\s*\r\?$##
:%s#\s*\r*$##
:%s/^\n\{3}//
:%s/^\n\+/\r/
:%s#<[^>]\+>##g
:'a,'bg/fred/s/dick/joe/igc
:%s= [^ ]\+=$&&=
:%s= \f\+=$&&=
:%s= \S\+=$&&
:s/(.*):(.*)\2 : \1/
:%s/^\(.*\)\n1$/\1/
:%s/^\{-}pdf/new.pdf/
:%s#\<[zy]\?tbl_[a-z-]\+>#\L&#gc
:%s///
:help /\{-}
:s/fred/a/g
:s/fred/\=@a/g
:%s/\f\+\.gif\>/\r&\r/g | v/\.gif$/d | %s/gif/jpg/
:%s/a/but/gie|:update|:next
:%s/suck\|buck/loopy/gc
:s/_date_/\=strftime("%c")/
:%s:\(\(\w\+\s\+)\)\{2}\)str1:\1str2:
:%s:\(\w\+)\)\(.*\s\+)\)\(\w\+)\)$:\3\2\1:
:%s/\d\+/\=(submatch(0)-3)/
:g/loc\|function/s/\d\+=submatch(0)+6/
:%s#txtdev\zs\d#\=submatch(0)+1#g
:%s/\(gg\)\@<=\d\+=submatch(0)+6/
:let i=10 | 'a,'bg/Abc/s/yy/\=i/ |let i=i+1
:let i=10 | 'a,'bg/Abc/s/xx\zsy\ze/\=i/ |let i=i+1
:%s/"\([^\]\+\.)*\zsxx/\1/
:nmap z :%s#\<=expand(" ")>#
:vmap z :%s/\<*\>/
```

1 TEXT MANIPULATION

general substitute command
delete dos returns ^M
turn dos returns ^M into real returns (fixes joined lines)
delete end of line blanks
same as above
clean both trailing spaces AND dos returns
same thing deleting empty lines
delete blocks of 3 empty lines
compressing empty lines
delete html tags, leave text
VERY USEFUL
duplicate end column
same as above
usually the same
reverse fields separated by :
delete duplicate lines
delete to 1st pdf only
lowercase with optional leading characters
delete possibly multi-line comments
help non-greedy
sub "fred" with contents of register "a"
better alternative as register not displayed
multiple commands on one line
then use @: to repeat
ORing (must break pipe)
insert datestring
working with columns sub any str1 in col3
swapping first and last column (4 columns)
decrement numbers by 3
increment numbers by 6 on certain lines only
better version of above
increment only numbers gg\d\d by 6 (another way)
convert yy to 10,11,12 etc
convert xxyy to xx11,xx12,xx13 (more precise)
find replacement text, use \zs to simplify substitute
pull word under cursor into LHS of a substitute
pull visually highlighted text into LHS of a substitute

1.2.1 Filter all form elements into paste register

```
:redir @*|sil exec 'g#<\(input\|select\|textarea\|/\=form\)\>#p'|redir END
:nmap ,z :redir @*<Bar>sil exec
\ 'g@<\(input\<Bar>select\<Bar>textarea\<Bar>/\=form\)\>#p'<Bar>redir END<CR>
```

1.2.2 Substutue within substitution

```
:%s,\(all/.*\)\@<=/,,-,g
:s#all/\zs.*#\=substitute(submatch(0), '/', '-', 'g')#
```

replace all / with - AFTER "all/"
same thing

1.3 Global command display

1 TEXT MANIPULATION

| | |
|---|---|
| <pre>:s#all/#&^M# s#/#-#g -j! :%s/.*\/\='cp '.submatch(0).' all/'.substitute(submatch(0),' ','-', 'g')/</pre> | <pre>sub by splitting line, re-joining sub inside sub</pre> |
|---|---|

1.2.3 Substituting a visual area

| | |
|---------------------------------------|--|
| <pre>:<,>s/Emacs/Vim/g gv</pre> | <pre>remember you DONT type the '<.>' re-select the previous visual area (ULTRA)</pre> |
|---------------------------------------|--|

1.3 Global command display

| | |
|---|---|
| <pre>:g/gladiolli/# :g/fred.*joe.*dick/ :g/\<fred\>/ :g/^\s*/d :g!/^dd/d :v/^dd/d :g/fred/,/joe/d :g/-----/-10,d :g/{/ ,/}- s/\n\+/\r/g :v\S/d :v./././-j :g/^\$/./-j :g/<input\ <form/p :g/^/put_ :g/^/m0 :'a,'b/^/m'b :g/^/t. :g/fred/t\$:g/stage/t'a :g/\(^I[^I]*\)\{80}/d :'a,'bg/somestr/co/otherstr/ :'a,'bg/str1/s/str1/&&&/ mo/str2/ :%norm jdd :.,\$g/^d/exe "norm! \" :'a,'bg/^d\+/\norm! ^A :g/fred/y A :g/fred/y A :let @*=@a :let @a=" g/Barratt/y A :let @*=@a :'a,'b g/^Error/ . w >> errors.txt :g/./yank put -1s/'/" /g s/.*/Print '&'/ :g/^MARK\$/r tmp.ex -d :g/z#.5 :g/z#.5 echo "===== :g/ /norm 2f r* :nmap :redir @a:g//:redir END:new:put! a</pre> | <pre>display with line numbers (YOU WANT THIS!) display all lines fred,joe & dick display all lines fred but not freddy delete all blank lines delete lines not containing string delete lines not containing string not line based (very powerfull) delete string & 10 previous lines delete empty lines but only between {...} delete empty lines (both types) compress empty lines compress empty lines ORing double space file (pu = put) reverse file (m = move) reverse a section a to b duplicate every line copy lines matching fred to EOF copy lines matching stage to marker a delete all lines containing at least 80 tabs match all lines containing "somestr" between markers a & b as above but also do a substitution delete every other line increment numbers increment numbers append all lines fred to register a (empty reg a first with qa.) put into paste buffer write out to errors.txt duplicate every line in a file wrap a print " around each duplicate replace string with contents of a file, -d deletes the "mark" display with context display beautifully replace 2nd with a star send output of previous global command to a new window</pre> |
|---|---|

1.4 Global combined with substitute (power editing)

| | |
|--|---|
| <code>:a,'bg/fred/s/joe/susan/gic</code> | can use memory to extend matching |
| <code>:g/fred/,/joe/s/fred/joe/gic</code> | non-line based (ultra) |
| <code>:/fred;/joe/-2,/sid/+3s/sally/alley/gIC</code> | find fred before beginning search for joe |

1.5 Changing case

| | |
|--|---|
| <code>guu</code> | lowercase line |
| <code>gUU</code> | uppercase line |
| <code>Vu</code> | lowercase line |
| <code>VU</code> | uppercase line |
| <code>g~~</code> | flip case line |
| <code>vEU</code> | upper case word |
| <code>vE~</code> | flip case word |
| <code>ggguG</code> | lowercase entire file |
| <code>vmap ,c :s/\<\(..\)\(\k*\)\>/\u\1\L\2/g</code> | titlise visually selected text (map for .vimrc) |
| <code>:%s/[!?!]_s\+ \a/U&\E/g</code> | uppercase first letter of sentences |
| <code>g<C-G></code> | count words in text file |

1.6 Reformatting text

| | |
|--|--|
| <code>gq}</code> | format a paragraph |
| <code>gqap</code> | format a paragraph |
| <code>ggVGgq</code> | reformat entire file |
| <code>Vgq</code> | current line |
| <code>:s/\{,69\};\s*\ .\{,69\}\s\+/\&\r/g</code> | break lines at 70 chars, if possible after a ';' |

1.7 Deletion without destroying buffer

| | |
|-------------------|-----------------------------|
| <code>"_d</code> | what you've ALWAYS wanted |
| <code>"_dw</code> | delete word (use blackhole) |

1.8 Essential

1.8 Essential

```
* # g* g#  
%  
.  
@:  
<C-N><C-P>  
<C-X><C-L>  
/<C-R><C-W>  
/<C-R><C-A>  
:set ignorecase  
:syntax on  
:h regexp  
:nmap ,s :source $VIM/_vimrc  
:nmap ,v :e $VIM/_vimrc  
:vmap sb "zdi<b><C-R>z</b><ESC>  
:vmap st "zdi<?= <C-R>z ?><ESC>
```

1 TEXT MANIPULATION

```
find word under cursor () (forwards/backwards)  
match brackets and tags {}, [], (), etc.  
repeat last modification  
repeat last : command (then @@)  
word completion in insert mode  
line complete SUPER USEFUL  
pull onto search/command line  
pull onto search/command line  
you nearly always want this  
colour syntax in perl, HTML, PHP etc.  
list all help topics containing regexp (TAB to step through list)  
read from vimrc  
open and edit local vimrc  
wrap <b></b> around VISUALLY selected text  
wrap <?= ?> around VISUALLY selected text
```

2 File Manipulation

2.1 Exploring

| | |
|-------------------------------|--|
| :Exp(lore) | file explorer (note: capital E) |
| :Sex(plore) | file explorer in split window |
| :ls | list of buffers |
| :cd .. | move to parent directory |
| :args | list of files |
| :lcd %:p:h | change to directory of current file |
| :autocmd BufEnter * lcd %:p:h | change to directory of current file automatically ¹ (put in .vimrc) |
| \be | buffer explorer list of buffers |
| \bs | buffer explorer (split window) |

2.2 Opening files & other tricks

| | |
|--------------------|--|
| gf | open file name under cursor (SUPER) |
| :nnoremap gF :view | open file under cursor, create if necessary |
| ga | display hex,ascii value of char under cursor |
| ggVGg? | rot13 whole file |
| ggg?G | rot13 whole file (quicker for large file) |
| :8 normal VGg? | rot13 from line 8 |
| :normal 10GVGg? | rot13 from line 8 |
| , | increment, decrement number under cursor |
| =5*5 | insert 25 into text (mini-calculator) |
| :e main_ | tab completes |
| main_ | include NAME of file in text (insert mode) |

2.3 Multiple files management

| | |
|---------------|--|
| :bn | goto next buffer |
| :bp | goto previous buffer |
| :wn | save file and move to next (super) |
| :wp | save file and move to previous |
| :bd | remove file from buffer list (super) |
| :bun | buffer unload (remove window but not from list) |
| :badd file.c | file from buffer list |
| :b 3 | go to buffer 3 |
| :b main | go to buffer with main in name eg main.c (ultra) |
| :sav php.html | save current file as php.html and "move" to php.html |
| :sav! %<.bak | save current file to alternative extension (old way) |
| :sav! %:r.cfm | save current file to alternative extension |

¹Script required: bufexplorer.vim http://www.vim.org/script.php?script_id=42

| | |
|--------------------------|---|
| :sav %s/fred/joe/ | do a substitute on file name |
| :sav %s/fred/joe/:r.bak2 | do a substitute on file name & ext. |
| !mv % %:r.bak | rename current file (DOS use rename or del) |
| :e! | return to unmodified file |
| :w c:/aaa/% | save file elsewhere |
| :e # | edit alternative file (also cntrl-^) |
| :rew | return to beginning of edited files list (:args) |
| :brew | buffer rewind |
| :sp fred.txt | open fred.txt into a split |
| :sball,:sb | split all buffers (super) |
| :scrollbind | in each split window |
| :map <F5> :ls<CR>:e # | pressing F5 lists all buffers, just type number |
| :set hidden | allows to change buffer w/o saving current buffer |

2.4 File-name manipulation

| | |
|-----------------------|-------------------------------|
| :h filename-modifiers | help |
| :w % | write to current file name |
| :w %:r.cfm | change file extension to .cfm |
| !echo %:p | full path & file name |
| !echo %:p:h | full path only |
| <C-R>% | insert filename (insert mode) |
| "%p | insert filename (normal mode) |
| /<C-R>% | search for file name in text |

2.5 Command over multiple files

| | |
|------------------------|-------------------------------|
| :argdo %s/foo/bar/e | operate on all files in :args |
| :bufdo %s/foo/bar/e | |
| :windo %s/foo/bar/e | |
| :argdo exe '%!sort' w! | include an external command |

2.6 Sessions (set of files)

| | |
|---|---|
| gvim file1.c file2.c lib/lib.h lib/lib2.h | load files for "session" |
| :mksession | create a session file (default session.vim) |
| gvim -S Session.vim | reload all files |

2.7 Modelines

| | |
|------------------------------|-----------------------------------|
| vim:noai:ts=2:sw=4:readonly: | makes readonly |
| vim:ft=html: | says use HTML syntax highlighting |
| :h modeline | help with modelines |

2.7.1 Creating your own GUI Toolbar entry

```
amenu Modeline.Insert\ a\ VIM\ modeline
\ <esc><esc>gg0vim:ff=unix ts=4 ss=4<CR>vim60:fdm=marker<esc>gg
```

2.8 Markers & moving about

| | |
|--------------------|--|
| .' | jump to last modification line (SUPER) |
| ‘. | jump to exact spot in last modification line |
| g; | cycle through recent changes (oldest first) ² |
| g, | reverse direction ³ |
| :changes | show entire list of changes |
| :h changelist | help for above |
| <C-O> | retrace your movements in file (starting from most recent) |
| <C-I> | retrace your movements in file (reverse direction) |
| :ju(mps) | list of your movements |
| :help jump-motions | explains jump motions |
| :history | list of all your commands |
| :his c | commandline history |
| :his s | search history |
| q/ | search history window (puts you in full edit mode) |
| q: | commandline history window (puts you in full edit mode) |
| : | history Window |

2.9 Editing/moving within insert mode

| | |
|-----------------------------|------------------------------------|
| <C-U> | delete all entered |
| <C-W> | delete last word |
| <HOME><END> | beginning/end of line |
| <C-LEFTARROW><C-RIGHTARROW> | jump one word backwards/forwards |
| <C-X><C-E>,<C-X><C-Y> | scroll while staying put in insert |

²(new in vim 6.3)

³(new in vim 6.3)

2.10 Abbreviations & maps

| | |
|---|--|
| <pre> :map <f7> : 'a,'bw! c:/aaa/x :map <f8> :r c:/aaa/x :map <f11> :.w! c:/aaa/xr<CR> :map <f12> :r c:/aaa/xr<CR> :ab php :map , set wak=no <CR> <ESC> <BACKSPACE> <LEADER> <BAR> <SILENT> iab phpdb exit("<hr>Debug <C-R>a "); </pre> | <pre> list of abbreviations beginning php list of maps beginning , allow use of F10 for win32 mapping (:h winaltkeys) enter escape backspace backslash execute quietly yank all variables into register a </pre> |
|---|--|

2.10.1 Display RGB colour under the cursor eg #445588

```

:nmap <leader>c :hi Normal guibg=#<c-r>=expand("<cword>")<cr><cr>

```

3 Registers

3.1 List your registers

| | |
|---|---|
| <code>:reg</code> | display contents of all registers |
| <code>:reg a</code> | display content of individual registers |
| <code>"1p...</code> | retrieve numeric registers one by one |
| <code>:let @y='yy@'</code> | pre-loading registers (put in .vimrc) |
| <code>qqq</code> | empty register "q" |
| <code>:let @a=@_</code> | clear register a |
| <code>:let @a=""</code> | clear register a |
| <code>:let @*=@a</code> | copy register a to paste buffer |
| <code>map <F11> "qyy:let @q=@q."zzz"</code> | |

3.2 Appending to registers

Yank 5 lines into "a" then add a further 5

```
"a5yy
10j
"A5yy
```

3.3 Using a register as a map (preload registers in .vimrc)

| | |
|-----------------------------------|--|
| <code>:let @m=":'a,'bs/'"</code> | |
| <code>:let @s=":%!sort -u"</code> | |

3.4 Redirection & paste register

| | |
|--|-----------------------------------|
| <code>:redir @*</code> | redirect commands to paste buffer |
| <code>:redir END</code> | end redirect |
| <code>:redir >> out.txt</code> | redirect to a file |
| <code>"*yy</code> | yank to paste |
| <code>"*p</code> | insert from paste buffer |
| <code>:'a,'by*</code> | yank range into paste |
| <code>:%y*</code> | yank whole buffer into paste |
| <code>:.y*</code> | yank current line to paste |
| <code>:nmap p :let @* = substitute(@*, '[[:print:]]', 'g')"</code> | filter non-printable characters |

3.4.1 Copy full path name

unix:

```
nnoremap <F2> :let @*=expand("%:p")<cr>
```

windows:

```
nnoremap <F2> :let @*=substitute(expand("%:p"), "/", "\\ ", "g")<cr>
```

3.5 Useful tricks

| | |
|---------------|--|
| "ayy@a | execute "vim command" in a text file |
| yy@" | same thing using unnamed register |
| u@. | execute command JUST typed in |
| :norm qqy\$jq | paste "normal commands" without entering insert mode |

4 Advanced

4.1 Command line tricks

| | |
|---|---|
| cat xx gvim - -c "v/^\\d\\d\\ ^ [3-9]/d " | filter a stream |
| ls gvim - | edit a stream! |
| gvim ftp://www.somedomain.com/index.html | uses netrw.vim |
| gvim -h | help |
| gvim -o file1 file2 | open into a split |
| gvim -c "/main" joe.c | open joe.c & jump to "main" |
| gvim -c "%s/ABC/DEF/ge update" file1.c | execute multiple command on a single file |
| gvim -c "argdo %s/ABC/DEF/ge update" *.c | execute multiple command on a group of files |
| gvim -c "argdo /begin/+1,/end/-1g/^/d update" *.c | remove blocks of text from a series of files |
| gvim -s "convert.vim" file.c | automate editing of a file (ex commands in convert.vim) |
| gvim -u NONE -U NONE -N | load vim without .vimrc and plugins (clean vim) |
| gvim -c 'normal ggdG"*p' c:/aaa/xp | access paste buffer contents (put in a script/batch file) |
| gvim -c 's/^/= @*/ hardcopy! q!' | print paste contents to default printer |
| gvim -d file1 file2 | vimdiff (compare differences) |
| dp | "put" difference under cursor to other file |
| do | "get" difference under cursor from other file |
| :grep somestring *.php | internal grep creates a list of all matching files |
| :h grep | use :cn(ext) :cp(rev) to navigate list |

4.2 External programs

| | |
|-------------------------------|--|
| :r!ls.exe | reads in output of ls |
| !!date | same thing (but replaces/filters current line) |
| :%!sort -u | sort unique content |
| : 'a,'b!sort -u | as above |
| !1} sort -u | sorts paragraph (note normal mode!!) |
| map <F9> :w!:c:/php/php.exe % | run file through php |
| map <F2> :w!:perl -c % | run file through perl |
| :runtime! syntax/2html.vim | convert txt to html |

4.3 Recording

| | |
|-----------|--|
| qq | record to q |
| q | end recording |
| @q | to execute |
| @@ | to repeat |
| 5@@ | to repeat 5 times |
| "qp | display contents of register q (normal mode) |
| <ctrl-R>q | display contents of register q (insert mode) |
| "qdd | put changed contacts back into q |

| | | |
|-------------------|--|--|
| @q | | execute recording/register q |
| nnoremap] @l:wbd | | combining a recording with a map (to end up in command mode) |

4.3.1 Operating a Recording on a Visual BLOCK

1. define recording/register

```
qq:s/ to/ from/g^Mq
```

2. Define Visual BLOCK

```
V}
```

3. hit : and the following appears

```
:'<,>
```

4. Complete as follows

```
:'<,>norm @q
```

4.4 Quick jumping between splits

```
map <C-J> <C-W>j<C-W>_
```

```
map <C-K> <C-W>k<C-W>_
```

4.5 Visual mode basics

| | | |
|--------------|--|--|
| v | | enter visual mode |
| V | | visual mode whole line |
| <C-V> | | enter VISUAL BLOCK mode |
| gv | | reselect last visual area (ultra) |
| o | | navigate visual area |
| "*y | | yank visual area into paste buffer |
| V% | | visualise what you match |
| V}J | | join visual block (great) |
| V}gJ | | join visual block w/o adding spaces |
| 0<C-V>10j2ld | | delete first 2 characters of 10 successive lines |

4.6 vimrc essentials

| | | |
|--------------------------------------|--|---|
| set incsearch | | jumps to search word as you type |
| set wildignore=*.o,*.obj,*.bak,*.exe | | tab complete now ignores these |
| set shiftwidth=3 | | for shift/tabbing |
| set vb t_vb=. | | set silent (no beep!) |
| set browsedir=buffer | | make 'open directory' use current directory |

4.6.1 Launching IE

```
:nmap ,f :update<CR>:silent !start c:\progra~1\intern~1\iexplore.exe file:///p<CR>
:nmap ,i :update<CR>: !start c:\progra~1\intern~1\iexplore.exe <WORD><CR>
```

4.6.2 FTPing from vim

```
cmap ,r :Nread ftp://209.51.134.122/public_html/index.html
cmap ,w :Nwrite ftp://209.51.134.122/public_html/index.html
gvim ftp://www.somedomain.com/index.html # uses netrw.vim
```

4.6.3 Autocmd

| | |
|--|---|
| autocmd bufenter *.tex map <F1> :!latex % | programming keys depending on file type |
| autocmd bufenter *.tex map <F2> :!xdvi -hush %<.dvi& | |
| autocmd BufRead * silent! %s/[\r \t]\+\$// | |
| autocmd BufEnter *.php :%s/[\r \t]\+\$//e | |

launch xdvi with current file dvi
automatically delete whitespace, trailing dos returns
same but only for php files

4.7 Conventional shifting and indenting

| | |
|-----------------|----------------------------------|
| :’a,’b>> | conventional Shifting/Indenting |
| :vnoremap < <gv | |
| :vnoremap > >gv | |
| >i{ | |
| >a{ | |
| >% | |
| <% | visual shifting (builtin-repeat) |
| | visual shifting (builtin-repeat) |
| | block shifting (magic) |

4.8 Pulling objects onto command/search line

| | |
|---------------|--|
| <C-R><C-W> | pull word under the cursor into a command line or search |
| <C-R><C-A> | |
| <C-R>- | |
| <C-R>[0-9a-z] | |
| <C-R>% | |
| <C-R>=somevar | |

pull WORD under the cursor into a command line or search
pull small register (also insert mode)
pull named registers (also insert mode)
pull file name (also #) (also insert mode)
pull contents of a variable (eg :let spray=”ray[0-9]”)

4.9 Capturing output of current script

| | |
|------------------------|---|
| :new r!perl # | opens new buffer,read other buffer |
| :new! x.out r!perl # | same with named file |
| :new+read!ls | |
| :new +put q %!sort | create a new buffer, paste a register "q" into it, then sort new buffer |

4.10 Inserting DOS carriage returns

| | |
|---------------|---|
| :%s/\$/\&/g | that's what you type |
| :%s/\$/\&/g | for Win32 |
| :%s/\$/\^M&/g | what you'll see where ^M is ONE character |
| :set list | display "invisible characters" |

4.11 Perform an action on a particular file or file type

```
autocmd VimEnter c:/intranet/note011.txt normal! ggVGg?
autocmd FileType *.pl exec('set fileformats=unix')
```

Retrieving last command line command for copy & pasting into text

```
i<c-r>:
```

Retrieving last Search Command for copy & pasting into text

```
i<c-r>/
```

4.12 Inserting line number

```
:g/^/exec "s/^/".strpart(line(".").", 0, 4)
:s/^/\=strpart(line(".").", 0, 5)
:s/^/\=line(' '). ' '
```

4.13 Numbering lines

| | |
|---|---|
| :set number | show line numbers |
| :map <F12> :set number!<CR> | map to toggle line numbers |
| :%s/^/\=strpart(line(' ').",0,&ts) | |
| :'a,'b!perl -pne 'BEGIN{\$a=223} substr(\$_,2,0)=\$a++' | number lines starting from arbitrary number |
| qqmnYP'n^Aq | in recording q repeat with @q |
| :.,\$g/^d/exe "normal! \" | increment existing numbers to end of file |
| o23qqYpq40@q | generate a list of numbers 23-64 |

4.14 Advanced incrementing

```
let g:I=0
function! INC(increment)
    let g:I =g:I + a:increment
    return g:I
end function
```

4.14.1 Create list starting from 223 incrementing by 5 between markers a,b

```
:let I=223
:'a,'bs/^/\=INC(5)/
```

4.14.2 Create a map for INC

```
cab viminc :let I=223 \| 'a,'bs/$/\=INC(5)/
```

4.15 Digraphs (non alpha-numeric)

| | |
|-------------|--|
| :digraphs | display table |
| :h dig | help |
| i<C-K>e' | enters |
| i<C-V>233 | enters (Unix) |
| i<C-Q>233 | enters (Win32) |
| ga | View hex value of any character |
| ::yl/<C-R>" | Pull a non-ascii character onto search bar |

4.16 Complex vim

| | |
|---|----------------|
| :%s/\<\(on\ off\)\>/\=strpart("offon", 3 * ("off" == submatch(0)), 3)/g | swap two words |
| :vnoremap <C-X> <Esc>'. "gvP" P | swap two words |

4.17 Syntax highlighting

| | |
|-----------------------------------|--|
| :set syntax=perl | force Syntax coloring for a file that has no extension .pl |
| :set syntax off | remove syntax coloring (useful for all sorts of reasons) |
| :colorscheme blue | change coloring scheme (any file in ~vim/vim?/?/colors) |
| vim:ft=html: | force HTML Syntax highlighting by using a modeline |
| :syn match DoubleSpace " " | example of setting your own highlighting |
| :hi def DoubleSpace guibg=#e0e0e0 | sets the editor background |

4.18 Preventions and security

| | |
|---|--|
| <code>:set noma</code> (non modifiable) | prevents modifications |
| <code>:set ro</code> (Read Only) | protect a file from unintentional writes |
| <code>:X</code> | encryption (do not forget your key!) |

4.19 Taglist ⁴

| | |
|--------------------------|----------------------------------|
| <code>:Tlist</code> | display tags (list of functions) |
| <code><C-]></code> | jump to function under cursor |

4.20 Folding

| | |
|-------------------|-----------------------------|
| <code>zf}</code> | fold paragraph using motion |
| <code>v}zf</code> | fold paragraph using visual |
| <code>zf'a</code> | fold to mark |
| <code>zo</code> | open fold |
| <code>zc</code> | re-close fold |

4.21 Renaming files

Rename files without leaving vim

```
:r! ls *.c
:%s/\(.*\).c/mv & \1.bla
:w !sh
:q!
```

4.22 Reproducing lines

| | |
|--|--------------------------------------|
| <code>imap] @@@hhkyWjl?@@@P/@@@3s</code> | reproduce previous line word by word |
| <code>nmap] i@@@hhkyWjl?@@@P/@@@3s</code> | reproduce previous line word by word |

⁴Script required: taglist.vim (http://www.vim.org/scripts/script.php?script_id=273)

4.23 Reading MS-Word documents ⁵

```
:autocmd BufReadPre *.doc set ro
:autocmd BufReadPre *.doc set hlsearch!
:autocmd BufReadPost *.doc %!antiword "%"
```

4.24 Random functions**4.24.1 Save word under cursor to a file**

```
function! SaveWord()
    normal yiw
    exe ':!echo '.@0.' $$$>$ word.txt'
endfunction
```

4.24.2 Delete duplicate lines

```
function! Del()
    if getline(".") == getline(line(".") - 1)
        norm dd
    endif
endfunction
```

4.24.3 Columnise a CSV file for display

```
:let width = 20
:let fill=' ' | while strlen(fill) < width | let fill=fill.fill | endwhile
:%s/\([^;]*\);.=strpart(submatch(1).fill, 0, width)/ge
:%s/\s\+$//ge
```

4.24.4 Highlight a particular csv column

```
function! CSVH(x)
    execute 'match Keyword /\^([^\,]*\)\{'.a:x.'\}zs[^\,]*/'
    execute 'normal ^'.a:x.'f,'
endfunction
command! -nargs=1 Csv :call CSVH(<args>)
```

Call with :Csv 5 to highlight fifth column

4.25 Miscallaenous commands

| | |
|-----------------------|--|
| :scriptnames | list all plugins, _vimrcs loaded (super) |
| :verbose set history? | reveals value of history and where set |
| :function | list functions |
| :func SearchCompl | List particular function |

⁵Program required: Antiword (<http://www.winfield.demon.nl>)

4.26 Vim traps

In regular expressions you must backslash + (match 1 or more)

In regular expressions you must backslash | (or)

In regular expressions you must backslash ((group)

In regular expressions you must backslash { (count)

| | |
|------------------------|----------------------------------|
| /fred\+ / | matches fred/freddy but not free |
| /\ (fred\)\{2,3} / | note what you have to break |
| /codes\(\n\—\s\)*where | normal regexp |
| /\vcodes(\n s)*where | very magic |

4.27 Help

| | |
|--------------------------|---|
| :h quickref | vim quick reference sheet (ultra) |
| :h tips | vim's own tips help |
| :h visual<C-D><TAB> | obtain list of all visual help topics |
| :h ctrl<C-D> | list help of all control keys |
| :helpg uganda | grep help files use :cn, :cp to find next |
| :h :r | help for :ex command |
| :h CTRL-R | normal mode |
| :h /\r | what's \r in a regexp (matches a <CR>) |
| :h \\zs | double up backslash to find \zs in help |
| :h i_CTRL-R | help for say <C-R> in insert mode |
| :h c_CTRL-R | help for say <C-R> in command mode |
| :h v_CTRL-V | visual mode |
| :h tutor | vim tutor |
| <C-[>, <C-T> | Move back & forth in help history |
| gvim -h | vim command line help |
| :helptags /vim/vim64/doc | rebuild all *.txt help files in /doc |
| :help add-local-help | |

5 Fun

```
:h 42
:h holy-grail
:h!
vim -c ":%s*s*%Cyrnfr)fcbafbe[0enz(Zbbyranne%|:%s)[[()])~)Ig|norm Vg?"}
```


6 Contact

6.1 Author (David Rayner)

Please email any errors or further tips etc to [david\(at\)rayninfo.co.uk](mailto:david(at)rayninfo.co.uk)

Updated version at <http://www.rayninfo.co.uk/vimtips.html>

6.2 Maintainer (Gavin Gilmour)

Please email any comments, suggestions including spelling, grammatical and/or formatting issues with this document to [gavin\(at\)brokentrain.net](mailto:gavin(at)brokentrain.net)

Updated version available at: <http://gavin.brokentrain.net/projects/vimtips/vimtips.pdf>

6.3 Source

The source of this document is available at: <http://github.com/gaving/vimtips/>.

6.4 Links

| | |
|---|----------------------|
| http://www.vim.org/ | Official site |
| http://chronos.cs.msu.su/vim/newsgroup.html | Newsgroup and Usenet |
| http://groups.yahoo.com/group/vim | Specific newsgroup |
| http://u.webring.com/hub?ring=vim | VIM Webring |
| http://www.truth.sk/vim/vimbook-OPL.pdf | Vim Book |
| http://vimdoc.sourceforge.net/ | Searchable VIM Doc |
| http://www.faqs.org/faqs/editor-faq/vim/ | VIM FAQ |