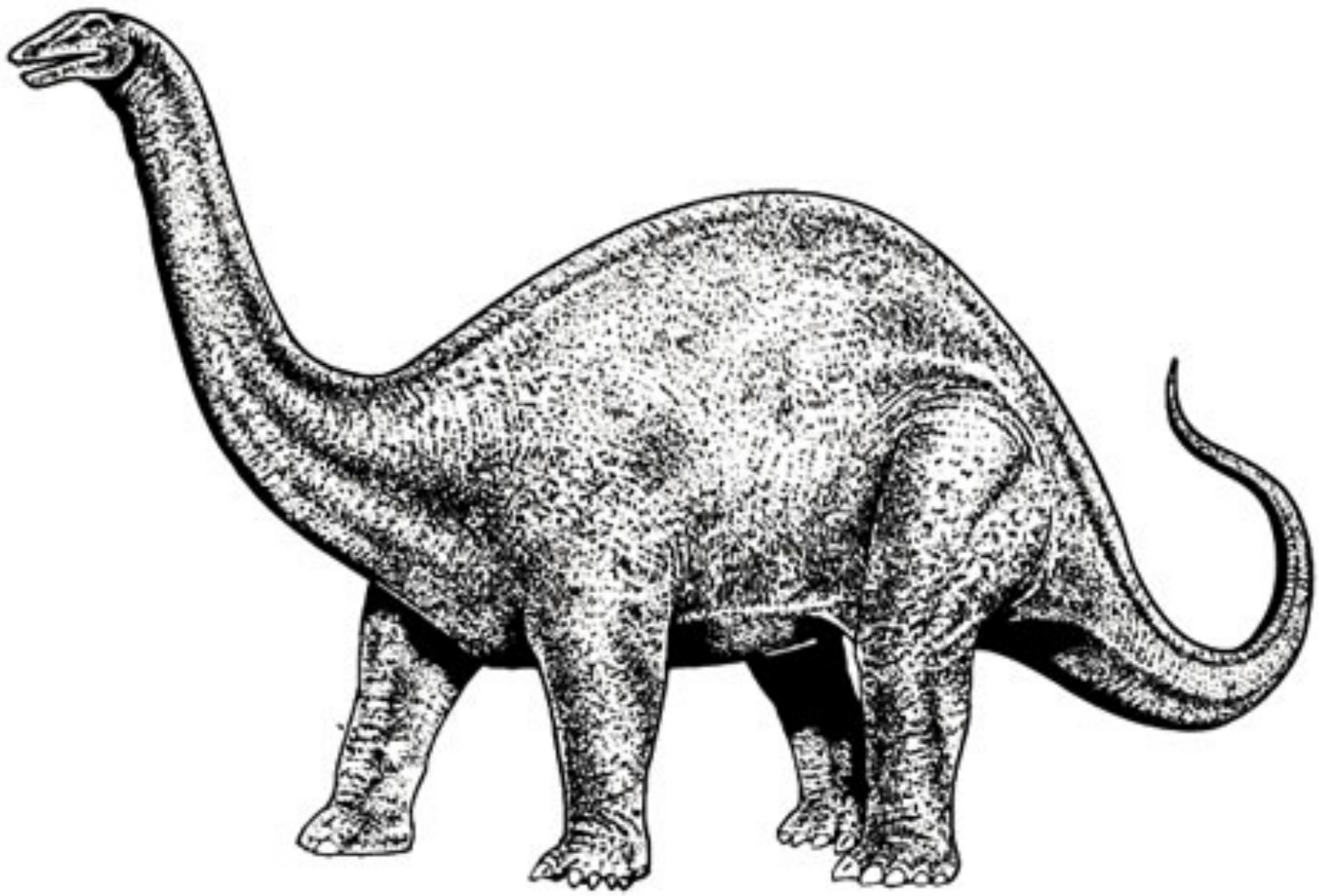
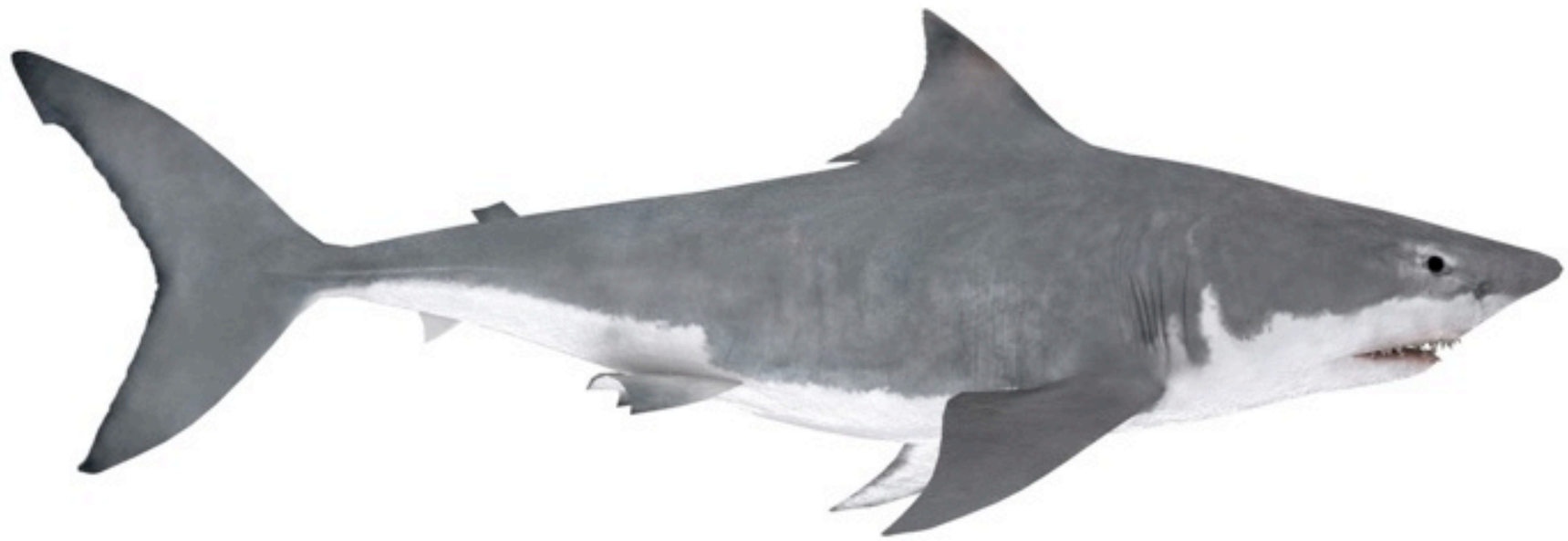
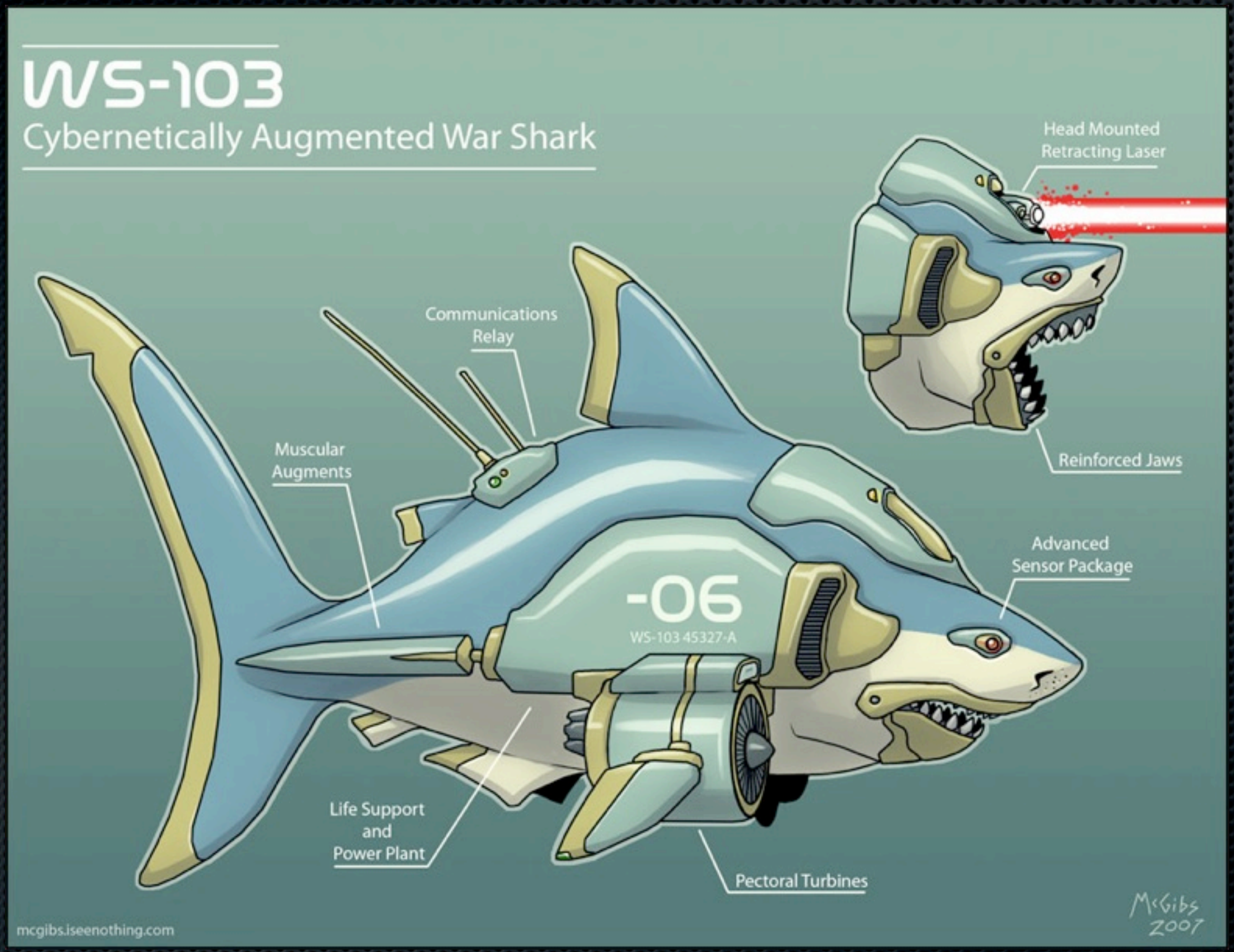
21st Century Vim

Tools, Techniques, and Configuration for the Vim Text Editor

Bill Odom (billodom@gmail.com) • *Strange Loop 2013*






WS-103
Cybernetically Augmented War Shark
Head Mounted
Retracting Laser
Communications
Relay
Muscular
Augments
Reinforced Jaws
Advanced
Sensor Package
-06
WS-103 45327-A
Life Support
and
Power Plant
Pectoral Turbines
mcgibs.iseenothing.com
McGibs
2007

WS-103

Cybernetically Augmented War Shark

Head Mounted
Retracting Laser

Communications
Relay

Reinforced Jaws

Advanced
Sensor Package

-06

WS-103 45327-A

Pectoral Turbines

McGibs
2007

A Quick History Lesson...

What is Vim?

Modal Text Editor

Built by Bram Moolenar...

/code/
?
Braugold

AMIGA

Based on `vi`...

...created by Bill Joy...

...in 1976.

1976

So why bother?

The Sales Pitch

Multi-Language

Terminal — bash — bodom — ttys000 — 180×55 — ⌘1

```
$ ls
2html.vim          cdrdaoconf.vim     diff.vim           gedcom.vim         java.vim           messages.vim       pf.vim             rpcgen.vim         splint.vim         uil.vim
README.txt         cdrtoc.vim         dircolors.vim      git.vim            javacc.vim         mf.vim             pfmain.vim         rpl.vim            spup.vim           updatedb.vim
a2ps.vim           cf.vim             diva.vim           gitcommit.vim      javascript.vim     mgl.vim            php.vim            rst.vim            spyce.vim          valgrind.vim
a65.vim            cfg.vim            django.vim         gitconfig.vim      jess.vim           mgp.vim            phtml.vim          rtf.vim            sql.vim            vb.vim
aap.vim            ch.vim             dns.vim            gitrebase.vim      jgraph.vim         mib.vim            pic.vim            ruby.vim           sqlanywhere.vim    vera.vim
abap.vim           change.vim         docbk.vim          gitsendemail.vim   jproperties.vim    mma.vim            pike.vim           samba.vim          sqlforms.vim       verilog.vim
abaqus.vim         changelog.vim      docbksgml.vim      gkrellmrc.vim      jsp.vim            mmix.vim           pilrc.vim          sas.vim            sqlinformix.vim    verilogams.vim
abc.vim            chaskell.vim       docbkxml.vim       gnuplot.vim        kconfig.vim        mmp.vim            pine.vim           sass.vim           sqlj.vim           vgrindefs.vim
abel.vim           cheetah.vim        dosbatch.vim       gp.vim             kix.vim            modconf.vim        pinfo.vim          sather.vim         sqloracle.vim      vhdl.vim
acedb.vim          chill.vim          dosini.vim         gpg.vim            kscript.vim        model.vim          plaintex.vim       scheme.vim         sqr.vim            vim.vim
ada.vim            chordpro.vim       dot.vim            grads.vim          kwt.vim            modsim3.vim        plm.vim            scilab.vim         squid.vim          viminfo.vim
aflex.vim          cl.vim             doxygen.vim        gretl.vim          lace.vim           modula2.vim        plp.vim            screen.vim         sshconfig.vim      virata.vim
ahdl.vim           clean.vim          dracula.vim        groff.vim          latte.vim          modula3.vim        plsql.vim          sd.vim             sshdconfig.vim     vmasm.vim
alsaconf.vim       clipper.vim        dsl.vim            groovy.vim         ld.vim             monk.vim           po.vim             sdc.vim            st.vim             voscm.vim
amiga.vim          cmake.vim          dtd.vim            group.vim          ldapconf.vim       moo.vim            pod.vim            sdl.vim            stata.vim          vrml.vim
aml.vim            cmusrc.vim         dtml.vim           grub.vim           ldif.vim           mp.vim             postscr.vim        sed.vim            stp.vim            vsejcl.vim
ampl.vim           cobol.vim          dtrace.vim         gsp.vim            lex.vim            mplayerconf.vim    pov.vim            sendpr.vim         strace.vim         wdiff.vim
ant.vim            coco.vim           dylan.vim          gtkrc.vim          lftp.vim           mrxvtrc.vim        povini.vim         sensors.vim        sudoers.vim        web.vim
antlr.vim          colortest.vim      dylanintr.vim      haml.vim           lhaskell.vim       msidl.vim          ppd.vim            services.vim       svn.vim            webmacro.vim
apache.vim         conaryrecipe.vim   dylanlid.vim       hamster.vim        libao.vim          msmessages.vim     ppwiz.vim          setserial.vim      syncolor.vim       wget.vim
apachestyle.vim    conf.vim           ecd.vim            haskell.vim        lifelines.vim      msql.vim           prescribe.vim      sgml.vim           synload.vim        whitespace.vim
arch.vim           config.vim         edif.vim           haste.vim          lilo.vim           mupad.vim          privoxy.vim        sgmldecl.vim       syntax.vim         winbatch.vim
art.vim            context.vim        eiffel.vim         hastepreproc.vim   limits.vim         mush.vim           procmail.vim       sgmllnx.vim        sysctl.vim         wml.vim
asm.vim            cpp.vim            elf.vim            hb.vim             lisp.vim           muttrc.vim         progress.vim       sh.vim             tads.vim           wsh.vim
asm68k.vim         crm.vim            elinks.vim         help.vim           lite.vim           mysql.vim          prolog.vim         sicad.vim          tags.vim           wsml.vim
asmh8300.vim       crontab.vim        elmfilt.vim        hercules.vim       litestep.vim       named.vim          promela.vim        sieve.vim          tak.vim            wvdial.vim
asn.vim            cs.vim             erlang.vim         hex.vim            loginaccess.vim    nanorc.vim         protocols.vim      simula.vim         takcmp.vim         xbl.vim
aspperl.vim        csc.vim            eruby.vim          hitest.vim         logindefs.vim      nasm.vim           psf.vim            sinda.vim          takout.vim         xdefaults.vim
aspvbs.vim         csh.vim            esmtprc.vim        hog.vim            logtalk.vim        nastran.vim        ptcap.vim          sindacmp.vim       tar.vim            xf86conf.vim
asterisk.vim       csp.vim            esqlc.vim          hostconf.vim       lotos.vim          natural.vim        purifylog.vim      sindaout.vim       taskdata.vim       xhtml.vim
asteriskvm.vim     css.vim            esterel.vim        html.vim           lout.vim           ncf.vim            pyrex.vim          sisu.vim           taskedit.vim       xinetd.vim
atlas.vim          cterm.vim          eterm.vim          htmlcheetah.vim    lpc.vim            netrc.vim          python.vim         skill.vim          tasm.vim           xkb.vim
autohotkey.vim     ctrlh.vim          eviews.vim         htmldjango.vim     lprolog.vim        netrw.vim          qf.vim             sl.vim             tcl.vim            xmath.vim
autoit.vim         cuda.vim           exim.vim           htmlm4.vim         lscript.vim        nosyntax.vim       quake.vim          slang.vim          tcsh.vim           xml.vim
automake.vim       cupl.vim           expect.vim         htmlos.vim         lsl.vim            nqc.vim            r.vim              slice.vim          terminfo.vim       xmodmap.vim
ave.vim            cuplsim.vim        exports.vim        ia64.vim           lss.vim            nroff.vim          racc.vim           slpconf.vim        tex.vim            xpm.vim
awk.vim            cvs.vim            fasm.vim           ibasic.vim         lua.vim            nsis.vim           radiance.vim       slpreg.vim         texinfo.vim        xpm2.vim
ayacc.vim          cvsrc.vim          fdcc.vim           icemenu.vim        lynx.vim           objc.vim           ratpoison.vim      slpspi.vim         texmf.vim          xquery.vim
b.vim              cweb.vim           fetchmail.vim      icon.vim           m4.vim             objcpp.vim         rc.vim             slrnrc.vim         tf.vim             xs.vim
baan.vim           cynlib.vim         fgl.vim            idl.vim            mail.vim           ocaml.vim          rcs.vim            slrnsc.vim         tidy.vim           xsd.vim
basic.vim          cynpp.vim          flexwiki.vim       idlang.vim         mailaliases.vim    occam.vim          rcslog.vim         sm.vim             tilde.vim          xslt.vim
bc.vim             d.vim              focexec.vim        indent.vim         mailcap.vim        omnimark.vim       readline.vim       smarty.vim         tli.vim            xxd.vim
bdf.vim            dcd.vim            form.vim           inform.vim         make.vim           openroad.vim       rebol.vim          smcl.vim           tpp.vim            yacc.vim
bib.vim            dcl.vim            forth.vim          initex.vim         man.vim            opl.vim            registry.vim       smil.vim           trasys.vim         yaml.vim
bindzone.vim       debchangelog.vim   fortran.vim        initng.vim         manconf.vim        ora.vim            remind.vim         smith.vim          trustees.vim       z8a.vim
blank.vim          debcontrol.vim     foxpro.vim         inittab.vim        manual.vim         pamconf.vim        resolv.vim         sml.vim            tsalt.vim          zsh.vim
bst.vim            debsources.vim     framescript.vim    ipfilter.vim       maple.vim          papp.vim           reva.vim           snnsnet.vim        tsscl.vim
btm.vim            def.vim            freebasic.vim      ishd.vim           masm.vim           pascal.vim         rexx.vim           snnspat.vim        tssgm.vim
bzr.vim            denyhosts.vim      fstab.vim          iss.vim            mason.vim          passwd.vim         rhelp.vim          snnsres.vim        tssop.vim
c.vim              desc.vim           fvwm.vim           ist.vim            master.vim         pcap.vim           rib.vim            snobol4.vim        uc.vim
calendar.vim       desktop.vim        fvwm2m4.vim        jal.vim            matlab.vim         pccts.vim          rnc.vim            spec.vim           udevconf.vim
catalog.vim        dictconf.vim       gdb.vim            jam.vim            maxima.vim         pdf.vim            rnoweb.vim         specman.vim        udevperm.vim
cdl.vim            dictdconf.vim      gdmo.vim           jargon.vim         mel.vim            perl.vim           robots.vim         spice.vim          udevrules.vim
$
```

Multi-Interface

Terminal — vim — bodom — ttys000 — 180×55 — ⌘1

```
" Press ? for help

.. (up a dir)
/Users/bodom/.env/.vim/
|+after/
|+colors/
|+compiler/
|+doc/
|+ftplugin/
|+plugin/
|+ruby/
|+syntax/
|-diff_with_saved.vim
|-filetype.vim
|-insert_matching_spaces.vim
|-map_movement_keys.vim
|-map_text_autoformat_keys.vim
|-paste_into_c_string.vim
|-redir_messages.vim
|-search_enhancements.vim
|-search_with_skip.vim
|-set_abbreviations.vim
|-switch_quotes.vim
`-toggle_highlights.vim
~
~
~
```

MacVim File Edit Tools Syntax Buffers Window Help Sun 15:09

UTF-8 (~/temp) - VIM

Hiragana
Katakana
CJK Unified Ideographs
Hangul Syllables
Arrows
Mathematical Operators
CJK Symbols and Punctuation

MMBackend.m + (~/macvim/src/MacVim) - VIM1

+ MMBackend.m | MacVim.h | MMTextView.m

```
    // As a last resort, check if it is one of the system defined colors.
    // The keys in this dictionary are also lowercase with no whitespace.
    obj = [sysColorDict objectForKey:stripKey];
    if (obj) {
        NSColor *col = [NSColor performSelector:NSSelectorFromString(obj)];
        if (col) {
            float r, g, b, a;
            col = [col colorUsingColorSpaceName:NSCalibratedRGBColorSpace];
            [col getRed:&r green:&g blue:&b alpha:&a];
            return (((int)(r*255+.5f) & 0xff) << 16)
                 + (((int)(g*255+.5f) & 0xff) << 8)
                 + ((int)(b*255+.5f) & 0xff);
        }
    }

    return INVALCOLOR;
}

- (BOOL)hasSpecialKeyWithValue:(NSString *)value
{
    NSEnumerator *e = [[MMBackend specialKeys] objectEnumerator];
    id obj;

    while ((obj = [e nextObject])) {
        if ([value isEqual:obj])
            return YES;
    }

    return NO;
}

- (void)enterFullscreen:(int)fuoptions background:(int)bg
```

Multi-Platform

Session Edit View Bookmarks Settings Help

feedalizer.rb

```ruby
1 require "rss/maker"
2 require "open-uri"
3 require "htree"
4 require "htree-fix"
5
6 class Feedalizer
7   attr_reader :source
8
9   def initialize(url, &b
10    @source = parse_sour
11    @rss = RSS::Maker::R
12
13    feed.link = url # A
14
15    if block_given?
16      instance_eval &blo
17      output!
18    end
19  end
20
21  def feed
22    @rss.channel
23  end
24
25  def scrape_items(xpath
26    @source.each(xpath_q
27      yield @rss.items.new_item, element
28    end
29  end
30
31  # TODO: Add Tidy backend
32  def parse_source(url)
33    open(url) { |io| HTree(io).to_rexml.elements }
-- INSERT --                                  27,41          Top
```

normal.c (~/vim/vim6/src) - VIM

File Edit Tools Syntax Buffers Window Words Help

```
find_start_of_word(pos)
    pos_T       *pos;
{
    char_u      *line;
    int         cclass;
    int         col;

    line = ml_get(pos->lnum);
    cclass = get_mouse_class(line + pos->col);

    while (pos->col > 0)
    {
        col = pos->col - 1;
/\<pos\>                                        2689,19        34%
```

[No Name] - GVIM

File Edit Tools Syntax Buffers Window Help

every four lines of normal sized type, there is five lines of the sidenote. eg:

ze in ems)

[No Name] - GVIM

File Edit Tools Syntax Buffers Window Help

[No Name]

```
1
~
~
~
~
~
~
~
~
~
~
~
[No Name]
 help.txt       For Vim versic

                        VIM -

      Move around:  Use the cu
                    "j" to go
 Close this window:  Use ":q<En
    Get out of Vim:  Use ":qa!<

 Jump to a subject:  Position t
help.txt [Help][RO]
```

boot.ini = (c:\) - GVIM

File Edit Tools Syntax Buffers Window Help

boot.ini

```
1 [boot loader]
2 timeout=30
3 default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
4 [operating systems]
5 multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft Windows XP Profe>
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
c:\boot.ini [RO]                                    1,1           All
"c:\boot.ini" [readonly] 5L, 211C
```

MacVim File Edit Tools Syntax Buffers Plugin Window Help (100%) Sun 7:33 PM

MMFullscreenWindow.m (~/src/vim/MacVim/src/MacVim) – VIM

MMFullscreenWi... TODO

```
(outer.size.height - inner.size.height)/2);
```

MMWindowController.m (~/src/vim/MacVim/src/MacVim) – VIM1

MMWindowCont... MMTextStorage.m

```
- (id)initWithVimController:(MMVimController *)controller
{
#ifndef NSAppKitVersionNumber10_4  // needed for non-10.5 sdk
# define NSAppKitVersionNumber10_4 824
#endif
    unsigned styleMask = NSTitledWindowMask | NSClosableWindowMask
            | NSMiniaturizableWindowMask | NSResizableWindowMask
            | NSUnifiedTitleAndToolbarWindowMask;

    // Use textured background on Leopard or later (uncomment this on Tiger for
    // polished metal window).
    if (floor(NSAppKitVersionNumber) > NSAppKitVersionNumber10_4)
        styleMask |= NSTexturedBackgroundWindowMask;

    NSWindow *win = [[NSWindow alloc]
            initWithContentRect:NSMakeRect(0,0,480,360)
                      styleMask:styleMask
                        backing:NSBackingStoreBuffered
                          defer:YES];

    if ((self = [super initWithWindow:win])) {
        vimController = controller;

        // Window cascading is handled by MMAppController.
        [self setShouldCascadeWindows:NO];

        NSView *contentView = [win contentView];
        vimView = [[MMVimView alloc] initWithFrame:[contentView frame]
                                     vimController:vimController];
        [contentView addSubview:vimView];

        // Create the tabline separator (which may be visible when the tabline
        // is hidden).  See showTabBar: for circumstances when the separator
        // should be hidden.
        NSRect tabSepRect = [contentView frame];
        tabSepRect.origin.y = NSMaxY(tabSepRect)-1;
        tabSepRect.size.height = 1;
        tablineSeparator = [[NSBox alloc] initWithFrame:tabSepRect];

        [tablineSeparator setBoxType:NSBoxSeparator];
```

101,37 10%

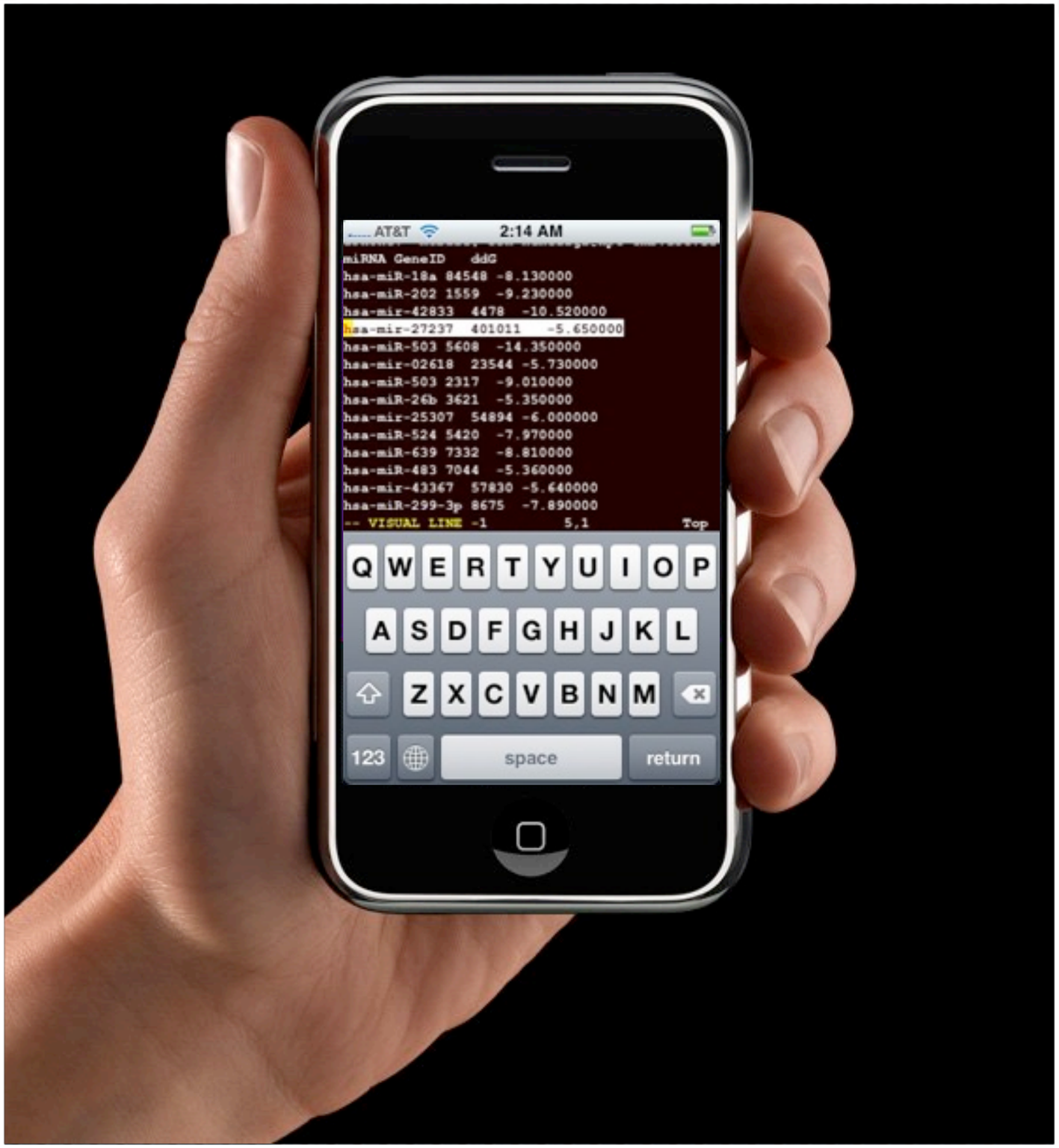
```
appear on demand
Suppressed, 0); //requires 10.3

en token;
ervation(.5, &token) == kCGErrorSuccess) {
5, kCGDisplayBlendNormal,
dColor, .0, .0, .0, true);

ate];

troller believe that it's now controlling us
owController will release target once in the
```

73,1 27%


AT&T
2:14 AM
miRNA GeneID ddG
hsa-miR-18a 84548 -8.130000
hsa-miR-202 1559 -9.230000
hsa-mir-42833 4478 -10.520000
hsa-mir-27237 401011 -5.650000
hsa-miR-503 5608 -14.350000
hsa-mir-02618 23544 -5.730000
hsa-miR-503 2317 -9.010000
hsa-miR-26b 3621 -5.350000
hsa-mir-25307 54894 -6.000000
hsa-miR-524 5420 -7.970000
hsa-miR-639 7332 -8.810000
hsa-miR-483 7044 -5.360000
hsa-mir-43367 57830 -5.640000
hsa-miR-299-3p 8675 -7.890000
-- VISUAL LINE -1 5,1 Top
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
123
space
return

SPELL A
OFF
GO
REPLAY
REPEAT
CLUE
MYSTERY SECRET
WORD
CODE
LETTER
SAY
IT
SPELL
ON
A B C D E F G H I J
K L M N O P Q R S T
U V W X Y Z
MODULE
SELECT
ERASE
ENTER
Speak & Spell
TEXAS INSTRUMENTS

MAGIC Etch A Sketch® SCREEN
Horizontal Dial
Vertical Dial

DECIVS MARIVS VENANTIVS
BASILIVS V C ET INL PRAEF
VRB PATRICIVS CONSVL
ORDINARIVS ARENAMET
PODIVM QVAE AB O M I
NANDI TERRAE MO
TVS RVINA PROS
TRAVIT SVMTV PRO
PRIORE STITVIT

```
#include <stdio.h>
#include <time.h>
int main ()
{
    time_t rawtime;
    struct tm *timeinfo;
    time(&rawtime);
    timeinfo = localtime ( &rawtime );
    int day = timeinfo->tm_mday;
    int month = timeinfo->tm_mon+1;
    int year = timeinfo->tm_year +1900;
    int age = year-1981;
    char *suffix;
    if(age % 10 == 1 && age != 11){
        suffix = (char*)"st";
    }else if(age %10== 2 && age !=12 ){
        suffix= (char*)"nd";
    }else if(age % 10 ==3 && age != 13 ){
        suffix= (char*)"rd";
    }else{
        suffix= (char*)"th";
    }
    if(month ==8 && day == 16){
        printf("Happy %d%s Birthday, Tim!\n", age,
        suffix);
    }else {
        printf("It's not your birthday..\n");
    }
    return 0;
```

Active Community

welcome home : vim online

http://www.vim.org/index.php

SPONSOR Vim development | VOTE for features | the editor | BUY the Vim book | HELP Uganda | LEARN Vim

not logged in (login)

Google™ Custom Search

Search

Home
Advanced search

About Vim
Community
News
Sponsoring
Trivia
Documentation
Download

Scripts
Tips
My Account

Site Help

What is Vim?

Vim is a highly configurable text editor built to enable efficient text editing. It is an improved version of the vi editor distributed with most UNIX systems. Vim is

News

Vim 7.3.003 is the current version

Vim 7.3 released!

[2010-08-15] Finally, after two y
includes an awful lot of bug f
before. For the details see th
page. (*Bram Moolenaar*)

Vim 7.3f BETA version ava

[2010-08-13] This is the last BE
report any problem you notic
have been! See the announc

Vim 7.3e BETA version ava

[2010-08-08] This might be the
problems have been fixed, it
in the announcement. (*Bram*

more news... Show you like

Recent Script Updates

[2010-08-18] Smart Parenthese
programming

Ads by Google

Python IDE
Faster, Easier

Vim Tips Wiki:Community Portal - Vim Tips Wiki

http://vim.wikia.com/wiki/Vim_Tips_Wiki:Community_Port

Vim Tips Wiki:Community Portal ...

wikia TECHNOLOGY

Create a new wiki | Wnodom | My Home | My talk | MORE | Log out

Edit this page | History | Move | Follow | Project page | Discussion

Search this wiki

Home
Community portal
To do
Top content
Folding category
Searching category
Create a new page
Upload a new photo
Recent changes | Random page
Staff blog | Special pages
Help

Latest activity
1,595 articles on this wiki
Building Vim
An hour ago by Fritzophrenic

Vim Tips Wiki:Community Portal [edit this page]

This is the starting point for all editors of the Vim Tips wiki.

- Please see our **policy introduction** for *how-to* information.
- To follow changes to this wiki, and to look for new events and problem edits, use **recent changes**.
- To discuss a tip, please edit the tip page and add to the Comments at the bottom (see the **discussion guidelines** for why we avoid the discussion pages for tips).
- On other pages (for example, this page), you are welcome to edit the **discussion page** to talk about the content.
- If you want help editing, you could start with **the tutorial** on the Central Wikia, or see our **editing help**.
- There are more help pages in the **help category**.
- And even more on **wikimedia.org**

More about the Vim community can be found at vim.org.

Contents [hide]

1 Asking questions
1.1 Creating a screenshot
2 Following (watching) changes
3 Downloading tips
4 Mirror sites

Asking questions [edit]

On this wiki, we discuss how to improve tips, but we don't discuss how to use Vim.

Questions about Vim should be sent to the **Vim mailing list**. The newsgroup comp.editors also

Ridiculously Powerful

Ridiculously Configurable

Open Source

Good for You

Good for Orphans

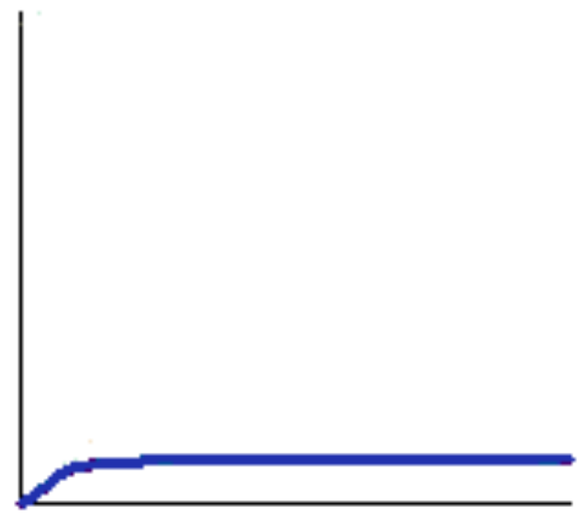
```
:help uganda
```

ensajja
olugalo

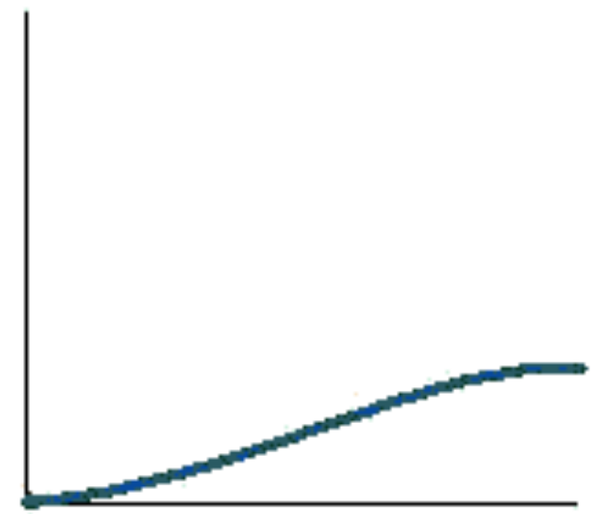
The Fine Print...

Classical learning curves for some common editors

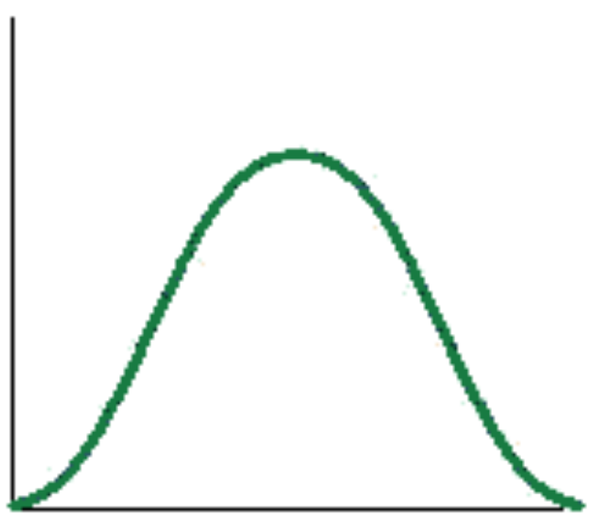
Notepad



pico



Visual Studio



vi

emacs

vi

and yet...

"i love vim" - Google Search

http://www.google.com/search?hl=en&safe=off&gbv=2&bi

⌘R DDG G Gm Gr Gt Gd Gv Subscribe... AWS Activity Twitter Search

"i love vim" - Google Search

Web Images Videos Maps News Shopping Gmail more ▼ BillOdom@gmail.c

Google

"i love vim" Search

Saf

About 116,000 results (0.15 seconds) Advanced search

Everything

zerosum dirt(nap) - Why I Love Vim

The Sales Pitch

The Sales Pitch ✓

So why bother?

So why bother?

- Save Time

So why bother?

- Save Time
- Increase Productivity

So why bother?

- Save Time
- Increase Productivity
- Reduce Tedium

So why bother?

- Save Time
- Increase Productivity
- Reduce Tedium
- Remain Sane

So why bother?

- Save Time
- Increase Productivity
- Reduce Tedium
- Remain Sane
- Have Fun

Use a single editor well.

The editor should be an extension of your hand; make sure your editor is configurable, extensible, and programmable.

—*The Pragmatic Programmer*
by Andrew Hunt and David Thomas

Goals

How to Get Started

A Few Big Ideas

Lots of Tips

Discussion, Questions Comments

Non-Goals

Editor war – Wikipedia, the free encyclopedia

http://en.wikipedia.org/wiki/Editor_war

editor wars

Editor war – Wikipedia, the free e...

New features Log in / create account

WIKIPEDIA
The Free Encyclopedia

Article Discussion Read Edit View history Search

Editor war

From Wikipedia, the free encyclopedia

For a type of conflict between wiki editors, see Edit war.

Editor war is the common name for the rivalry between users of the vi and Emacs text editors. The rivalry has become a lasting part of hacker culture and the free software community.

Many flame wars have been fought between groups insisting that their editor of choice is the paragon of editing perfection, and insulting the others. Unlike the related battles over operating systems, programming languages, and even source code indent style, choice of editor usually only affects oneself.

Contents [hide]

1 Differences between vi and Emacs
1.1 Benefits of vi-like editors
1.2 Benefits of Emacs
2 Humour
3 Current state of the editor war
4 References
5 External links

“Escape Meta Alt Control Shift”

“Emacs: A great operating system, lacking only a decent editor.”

vi has two modes:
“beep repeatedly” and
“break everything”

Sometimes, people ask me if it is a sin in the church of Emacs to use the editor Vi. It is true that Vi-Vi-Vi is the editor of the beast. But using a free version of Vi is not a sin but a penance.

— Richard Stallman (creator of EMACS)

Getting Started

Step One: Don’t

I

The Tutorial

1 Basic Editing
2 Editing a Little Faster
3 Searching
4 Text Blocks and Multiple Files
5 Windows
6 Basic Visual Mode
7 Commands for Programmers
8 Basic Abbreviations, Keyboard Mapping, and Initialization Files
9 Basic Command-Mode Commands
10 Basic GUI Usage
11 Dealing with Text Files
12 Automatic Completion
13 Autocommands
14 File Recovery and Command-Line Arguments
15 Miscellaneous Commands
16 Cookbook
17 Topics Not Covered

version 1.1
April 1st, 06

vi / vim graphical cheat sheet

Esc normal mode												
~ toggle case	! external filter	@• play macro	# prev ident	$ eol	% goto match	^ "soft" bol	& repeat :s	* next ident	(begin sentence	) end sentence	"soft" bol __ down	+ next line
`• goto mark	1 [2]	2	3	4	5	6	7	8	9	0 "hard" bol	- prev line	= auto[3] format
	Q ex mode	W next WORD	E end WORD	R replace mode	T• back 'till	Y yank line	U undo line	I insert at bol	O open above	P paste before	{ begin parag.	} end parag.
	q• record macro	w next word	e end word	r• replace char	t• 'till	y yank [1,3]	u undo	i insert mode	o open below	p paste[1] after	[• misc	]• misc
	A append at eol	S subst line	D delete to eol	F• "back" find ch	G eof/ goto ln	H screen top	J join lines	K help	L screen bottom	: ex cmd line	"• reg. [1] spec	| bol/ goto col
	a append	s subst char	d delete [1,3]	f• find char	g• extra[6] cmds	h ←	j ↓	k ↑	l →	; repeat t/T/f/F	'• goto mk. bol	\• not used!
	Z• quit [4]	X back-space	C change to eol	V visual lines	B prev WORD	N prev (find)	M screen mid'l	< un-[3] indent	> indent[3]	?• find (rev.)		
	Z• extra[5] cmds	x delete char	c change [1,3]	v visual mode	b prev word	n next (find)	m• set mark	, reverse t/T/f/F	. repeat cmd	/• find		

motion — moves the cursor, or defines the range for an operator

command — direct action command, if red, it enters insert mode

operator — requires a motion afterwards, operates between cursor & destination

extra — special functions, requires extra input

q• — commands with a dot need a char argument afterwards

bol = beginning of line, eol = end of line, mk = mark, yank = copy

words: `quux(foo, bar, baz);`
WORDs: `quux(foo, bar, baz);`

Main command line commands ('ex'):

:w (save), :q (quit), :q! (quit w/o saving)
:e f (open file f),
:%s/x/y/g (replace 'x' by 'y' filewide),
:h (help in vim), :new (new file in vim),

Other important comands:

CTRL-R: redo (vim),
CTRL-F/-B: page up/down,
CTRL-E/-Y: scroll line up/down,
CTRL-V: block-visual mode (vim only)

Visual mode:

Move around and type operator to act on selected region (vim only)

Notes:

(1) use "x before a yank/paste/del command to use that register ('clipboard') (x=a..z,*) (e.g.: "ay$ to copy rest of line to reg 'a')

(2) type in a number before any action to repeat it that number of times (e.g.: 2p, d2w, 5i, d4j)

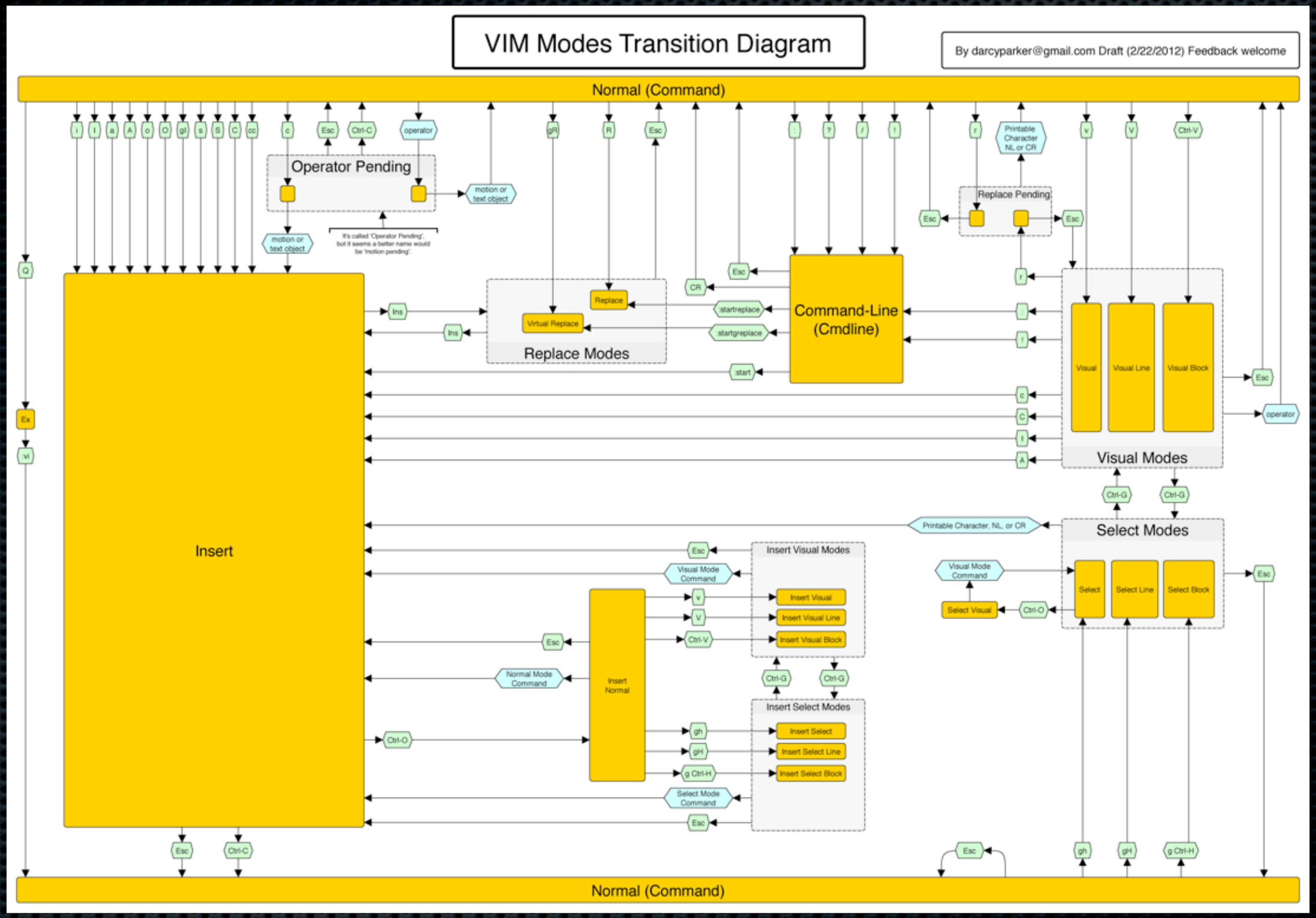
(3) duplicate operator to act on current line (dd = delete line, >> = indent line)

(4) ZZ to save & quit, ZQ to quit w/o saving

(5) zt: scroll cursor to top, zb: bottom, zz: center

(6) gg: top of file (vim only), gf: open file under cursor (vim only)

For a graphical vi/vim tutorial & more tips, go to **www.viemu.com** - home of ViEmu, vi/vim emulation for Microsoft Visual Studio


VIM Modes Transition Diagram
By darcyparker@gmail.com Draft (2/22/2012) Feedback welcome
Normal (Command)
Operator Pending
motion or text object
It's called 'Operator Pending', but it seems a better name would be 'motion pending'.
Replace Pending
Printable Character NL or CR
Insert
Replace Modes
Virtual Replace
Replace
Command-Line (Cmdline)
startreplace
startgreplace
Visual
Visual Line
Visual Block
Visual Modes
Select Modes
Select
Select Line
Select Block
Select Visual
Visual Mode Command
Printable Character, NL, or CR
Insert Visual Modes
Insert Visual
Insert Visual Line
Insert Visual Block
Insert Normal
Normal Mode Command
Insert Select Modes
Insert Select
Insert Select Line
Insert Select Block
Select Mode Command
Normal (Command)

A small piece of advice:
Don’t try to do it all at once.

Get a good version of vi that doesn't make you abandon everything you already know—then slowly, steadily build up your repertoire of magic tricks...

I've been using some form of vi for years, and I still learn new stuff all the time. Frankly, that's part of its appeal—it's as deep as I want to go, ***and there's more power every time I take a few minutes to learn it.***

– from an email exchange between Bill Odom and Merlin Mann

You should not try to learn every command an editor offers. That would be a complete waste of time. Most people only need to learn 10 to 20 percent of the commands for their work. ***But it's a different set of commands for everybody.***

—Bram Moolenar

I am amzedf at how mcuh easier it is to use vim when I jhave had a few beers. The modal commands just seem to flow from my fgingertips.

—Bill Odom (from “vim beer amazement.txt”)

Getting Vim

UNIX or Linux?

Done.

Mac OS X:

code.google.com/p/macvim/

macvim - Project Hosting on Google Code

http://code.google.com/p/macvim/

macvim – Project Hosting on Goo...

BillOdom@gmail.com | My favorites ▼ | Profile | Sign out

macvim

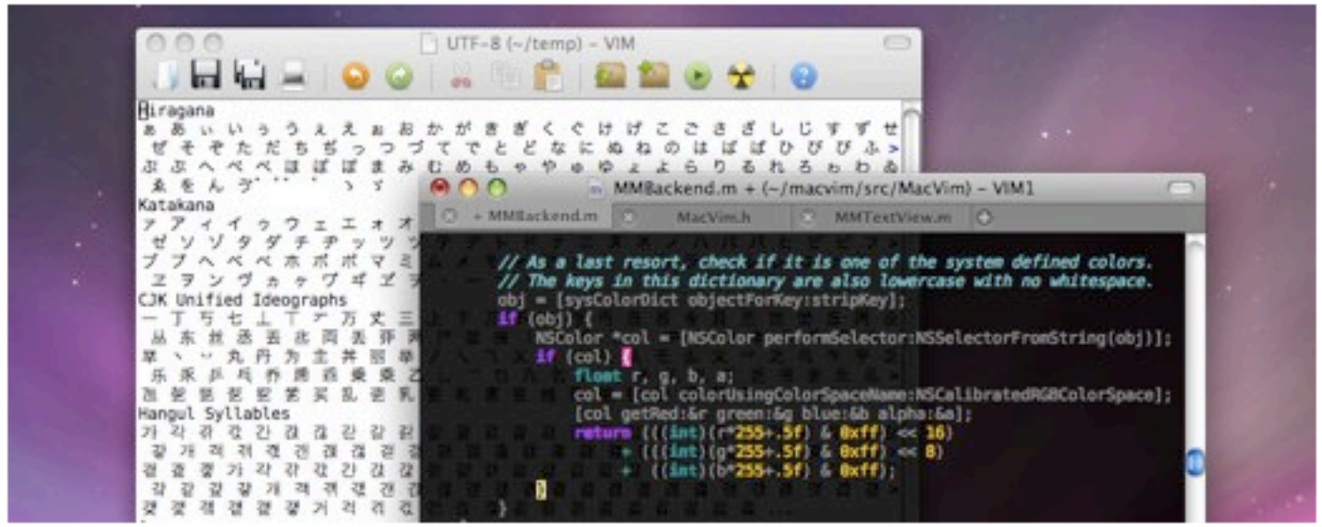
Vim for the Mac

Search projects

Project Home | **Downloads** | **Wiki** | **Issues** | **Source**

Summary | Updates | People

MacVim is a port of the text editor Vim to Mac OS X.



MacVim supports multiple windows with tabbed editing and a host of other features such as:

- bindings to standard OS X keyboard shortcuts (⌘Z, ⌘V, ⌘A, ⌘G, etc.),
- transparent backgrounds,

Starred (view starred projects)

Activity: High

Code license:
GNU General Public License v2

Labels:
vim, editor, mac, osx, cocoa, objective-c

Featured wiki pages:
ReportingBugs
Show all »

Blogs:
Björn's Blog

External links:
Source code repository

Everybody else:

www.vim.org/download.php

Getting Help

`:help` (or just `:h`)

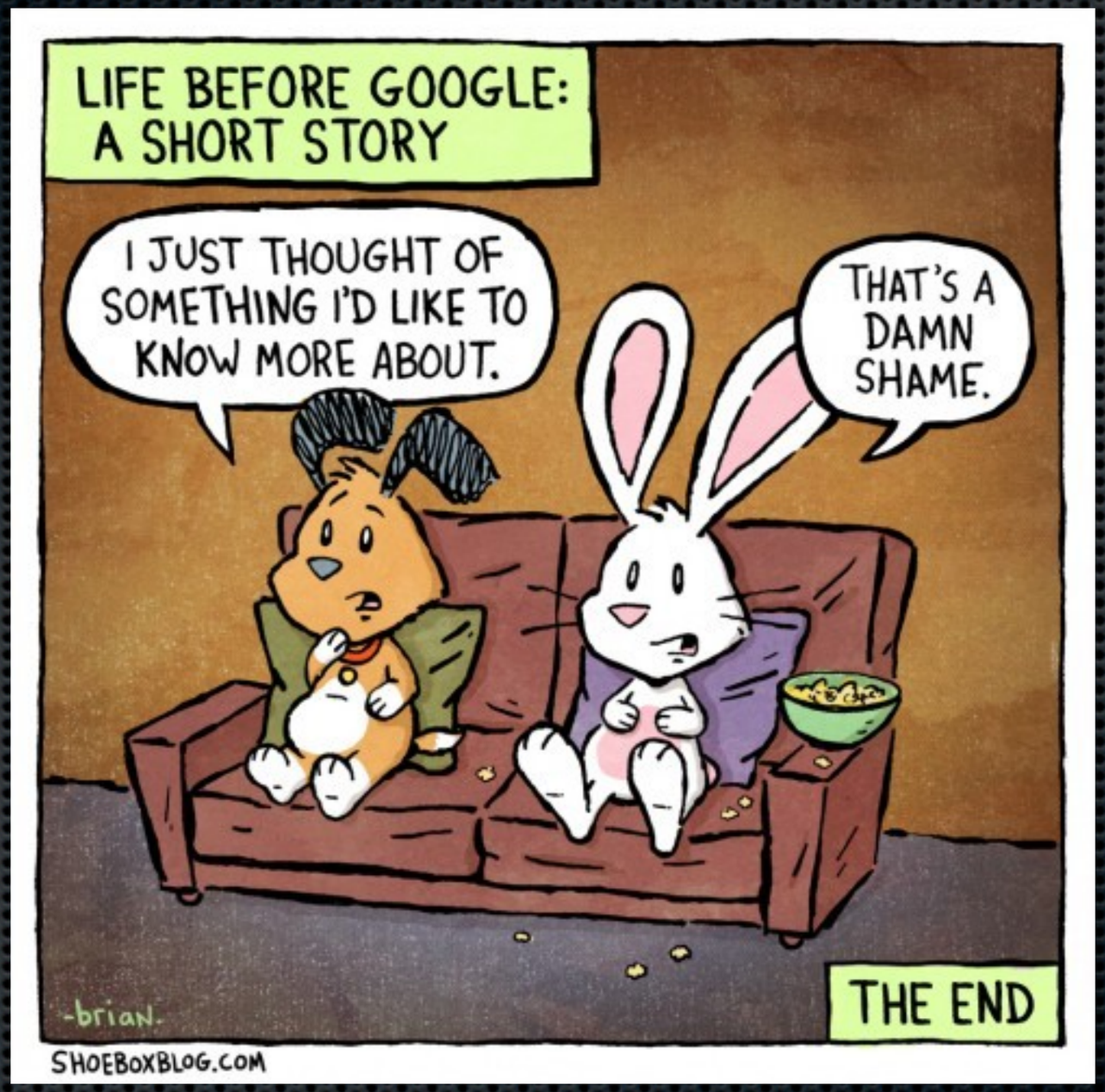
`:help :help`

`:helpgrep` (or just `:helpg`)

`:help!`

`:h 42`

`:h holy-grail`


LIFE BEFORE GOOGLE:
A SHORT STORY
I JUST THOUGHT OF SOMETHING I'D LIKE TO KNOW MORE ABOUT.
THAT'S A DAMN SHAME.
-brian-
THE END
SHOEBOXBLOG.COM

Modes

A Whole Lotta Modes

Insert Mode

Normal Mode

Command Mode

Visual Mode

Select Mode

Ex Mode

Operator-pending Mode

Replace Mode

Virtual Replace Mode

Insert Normal Mode

Insert Visual Mode

Insert Select Mode

A Whole Lotta Modes

Insert Mode

Normal Mode

Command Mode

Visual Mode

Insert Mode

- Typical text entry and basic editing
- Arrow keys (and mouse) work
- Lots of ways to get there...

Insert Mode

i a c o r s

I A C O R S

Insert Mode

- **`<Esc>`** to get out
- **`<C-O>`** to temporarily get out

Normal Mode

- Moving around
- More complex editing
- Default mode

Moving Around

```
k
h • l
j
```

Moving Around

```
H M L
gg
123G
:123
```

Moving Around

- Many more advanced motions available
- But the arrow keys & mouse still work!

Moving Around

gj

gk

Traveling Without Moving

Traveling Without Moving

zz

zt

zb

Traveling Without Moving

```
:let &scrolloff=999-&scrolloff
```

Instant Searches

- *
- #
- g*
- g#

Completion

- **<C-N>**
- **<C-P>**
- **<C-X><C-...>**

Command-Line Mode

- Ex commands
- Search commands
- Filter commands

Visual Mode

What?

Visual mode is a flexible and easy way to select a piece of text for an operator.

— `:help visual-mode`

Why use it?

Why use it?

- Simple way to select text and operate on it

Why use it?

- Simple way to select text and operate on it
- Often faster (and easier) than other methods

Why use it?

- Simple way to select text and operate on it
- Often faster (and easier) than other methods
- Sometimes it’s the only way to indicate an arbitrary block of text

Three Modes in One!

Three Modes in One!

- Visual

Three Modes in One!

- Visual
- Visual Line

Three Modes in One!

- Visual
- Visual Line
- Visual Block

Three Modes in One!

Three Modes in One!

- Visual - characterwise

Three Modes in One!

- Visual - characterwise
- Visual Line - linewise

Three Modes in One!

- Visual - characterwise
- Visual Line - linewise
- Visual Block - er... rectanglewise

Getting There

Getting There

- `v` for Visual

Getting There

- `v` for Visual
- `V` for Visual Line

Getting There

- **`v`** for Visual
- **`v`** for Visual Line
- **`<C-v>`** for Visual Block

Getting Out

Getting Out

Defining the Visual Area

Defining the Visual Area

- Enter Visual mode

Defining the Visual Area

- Enter Visual mode
- Move around

Defining the Visual Area

- Enter Visual mode
- Move around
- ...or use a text object

Defining the Visual Area

- Enter Visual mode
- Move around
- ...or use a text object
- ...or a search

Defining the Visual Area

- Enter Visual mode
- Move around
- ...or use a text obje
- ...or a search
- *...or even a mouse*

Doing Stuff

Doing Stuff

- Define a visual area

Doing Stuff

- Define a visual area
- Use an operator

Doing Stuff

- Define a visual area
- Use an operator
- ...or an Ex command

Visual Operators

Visual Operators

- All the usual suspects:

~ d c y > < ! = gq

Visual Operators

- All the usual suspects:

  ~ d c y > < ! = gq

- ...and a bunch more:

  r s C S R x D

  X Y p J U u ^] I

Oops, Wrong Mode

Oops, Wrong Mode

- No problem

Oops, Wrong Mode

- No problem
- Switch among modes with

  `v`, **`V`**, or **`<C-V>`**

Oops, Wrong Mode

- No problem
- Switch among modes with

  `v`, **`V`**, or **`<C-V>`**
- Start and end points are preserved

Oops, Wrong Spot

Oops, Wrong Spot

- Started in the wrong place?

Oops, Wrong Spot

- Started in the wrong place?
- Toggle between start end end with **o** (lowercase o)

Oops, Wrong Corner

Oops, Wrong Corner

- For Visual Blocks

Oops, Wrong Corner

- For Visual Blocks
- Need to move the other side?

Oops, Wrong Corner

- For Visual Blocks
- Need to move the other side?
- Switch to the other corner with **o** (uppercase O)

0.0

Prepending Text in a Block

Prepending Text in a Block

- Define a Visual Block

Prepending Text in a Block

- Define a Visual Block
- Press `I` (“Insert at beginning”)

Prepending Text in a Block

- Define a Visual Block
- Press `I` (“Insert at beginning”)
- Type the text to be prepended

Prepending Text in a Block

- Define a Visual Block
- Press `I` (“Insert at beginning”)
- Type the text to be prepended
- Hit Escape

Prepending Text in a Block

- Define a Visual Block
- Press `I` (“Insert at beginning”)
- Type the text to be prepended
- Hit Escape
- Works in the middle of a block, too

Appending Text in a Block

Appending Text in a Block

- Define a Visual Block

Appending Text in a Block

- Define a Visual Block
- Press **A** (“Append”)

Appending Text in a Block

- Define a Visual Block
- Press **`A`** (“Append”)
- Type the text to be appended

Appending Text in a Block

- Define a Visual Block
- Press **A** (“Append”)
- Type the text to be appended
- Hit Escape

Appending Text in a Block

Appending Text in a Block

- Start a Visual Block

Appending Text in a Block

- Start a Visual Block
- ...but press **$** to move to the end of a line

Appending Text in a Block

- Start a Visual Block
- ...but press **$** to move to the end of a line
- Press **A** (for “Append”)

Appending Text in a Block

- Start a Visual Block
- ...but press **$** to move to the end of a line
- Press **A** (for “Append”)
- Type the text to be appended

Appending Text in a Block

- Start a Visual Block
- ...but press **$** to move to the end of a line
- Press **A** (for “Append”)
- Type the text to be appended
- Hit Escape

Appending Text in a Block

- Start a Visual Block
- ...but press **$** to move to the end of a line
- Press **A** (for “Append”)
- Type the text to be appended
- Hit Escape
- Text is appended to each line, sensitive to the length

Example Maps

```
" Backspace in Visual mode deletes selection.
"
vnoremap <BS> d

" Tab/Shift+Tab indent/unindent the highlighted
" block (and maintain the highlight after changing
" the indentation). Works for both Visual and
" Select modes.
"
vnoremap <Tab>    >gv
vnoremap <S-Tab>  <gv
```

Example Maps

```
" Join all lines in a paragraph
"
nnoremap <S-F7>  vipJ

" Format selected lines
"
xnoremap Q  gq
```

Visual Key Map Gotchas

Visual Key Map Gotchas

- Watch out for **`:vmap`**
- Prefer **`:xmap`** and **`:smap`** where possible

Example Tricks

Example Tricks

- Count bytes in an arbitrary selection

Example Tricks

- Count bytes in an arbitrary selection
- See text objects

Example Tricks

- Count bytes in an arbitrary selection
- See text objects
- Draw boxes

Example Tricks

- Count bytes in an arbitrary selection
- See text objects
- Draw boxes
- Abuse indentation

Example Tricks

- Count bytes in an arbitrary selection
- See text objects
- Draw boxes
- Abuse indentation
- Filter a visual range

Important Settings

Important Settings

- virtualedit

Important Settings

- virtualedit
- selectmode

Important Settings

- virtualedit
- selectmode
- keymodel

Important Settings

- virtualedit
- selectmode
- keymodel
- mouse

Important Settings

```
:set virtualedit=block
:set selectmode=key,mouse
:set keymodel=startsel,stopsel
:set mouse=a
```

Useful Help Topics

```
:help visual-mode
:help v_…
:help v_b_…
```

Operators

c	gu	g?
d	gU	>
y	!	<
~	=	zf
g~	gq	g@

Operators

```
:help operator
```

Motions

```
h j k l                b B ge gE
gj gk                  - +
0 ^ $                  ( ) { }
g0 g^ g$ g_            ]] ][ [[
f F t T                | _ %
w W e E
```

Motions

```
:help motion.txt
```

Text Objects

w - Words

W - WORDs

s - Sentences

p - Paragraphs

t - Tagged blocks

Blocks delimited by

[ ] ( ) { } <>

Strings delimited by

" ' `

Text Objects

```
:help text-objects
```

Registers

What’s a Register?

What’s a Register?

- Stores a chunk of text

What’s a Register?

- Stores a chunk of text
- Like the system clipboard

What's a Register?

- Stores a chunk of text
- Like the system clipboard
- But much more flexible

The Unnamed Register

The Unnamed Register

- You already use this one

The Unnamed Register

- You already use this one
- The one you get if you don’t specify

The Unnamed Register

- You already use this one
- The one you get if you don’t specify
- Target of typical delete, yank, and put operations (a.k.a. cut, copy, and paste)

Simple Registers

- 26 named registers
- Accessed as `"a` through `"z`

Appending to Registers

Appending to Registers

- Deletes and yanks overwrite registers

Appending to Registers

- Deletes and yanks overwrite registers
- But they don’t have to

Appending to Registers

- Deletes and yanks overwrite registers
- But they don’t have to
- Uppercase the name to append instead

Numbered Registers

- You don’t always have to be explicit
- Vim tracks recent deletes
- Look in **"1** through **"9**

Other Registers

Current filename:

"%

Alternate filename:

"#

Other Registers

Last “small” delete:

"_

Last inserted text:

":

Other Registers

Last search:

`"/`

Last Ex command:

`":`

Other Registers

Expression:

"=

(Lots more on this one later)

Other Registers

System clipboard:

"*

System Selection (X11):

"+

Other Registers

Black hole:

"_

Viewing Registers

- View with **`:registers`**
  - Or just **`:reg`**
- View specific registers with **`:reg abc`**

Accessing Registers

Normal mode:

`"<register><operator>`

Accessing Registers

Visual mode:

`"<register><operator>`

Accessing Registers

Insert mode and command line:

```
<C-R><register>
<C-R><C-R><register>
<C-R><C-O><register>
<C-R><C-P><register>
```

Accessing Registers

Ex commands and scripts:

```
@<register>
```

Registers == Macros

Execute in Normal mode:

```
@<register>
```

Registers == Ex Commands?

Execute in Command mode:

```
:@<register>
```

Clearing Registers

Assign on the command line:

```
:let @a = ""
```

Record an empty macro:

```
qaq
```

Expression Register

- Value is the result of an expression
- Can be used interactively or in scripts
- Accessed as `"=` or `@=` or `\=` depending on context

Expression Register Example

```
inoremap <F2>
  <C-R>=strftime("%c")<CR>
```

Expression Register Example

```
:cnoremap <C-G>
    <C-R><C-R>=
    GetLiteralPattern()
    <CR>
```

Expression Register Example

```
:xnoremap \/
    "vy/
    <C-R><C-R>
    =StringToPattern(@v)
    <CR>
```

Tips & Tricks

Copy to the OS clipboard:

```
"*yy
```

Tips & Tricks

Paste from the OS clipboard:

`"*p`

(Can be more reliable than pasting through a terminal window.)

Tips & Tricks

Link unnamed register and OS clipboard:

```
:set clipboard=unnamed
```

Tips & Tricks

Pre-load in .vimrc:

```
:let @a = "123 Main Street"
:let @p = "800-555-1212"
```

Tips & Tricks

“Search” for text without moving cursor:

```
:let @/ = "stuff"
```

Tips & Tricks

Execute accidentally-inserted command:

```
<Esc>u:@.
```

Tips & Tricks

Edit a macro:

`"mp`

...edit macro text...

`"myy`

Tips & Tricks

Combine several registers:

```
:let @* = @a . @b . @c
```

Tips & Tricks

Capture output to a register:

```
:redir @A
```

Tips & Tricks

Yank matching lines into a register:

```
:g/pattern/y A
```

(Note uppercase register name)

Register Help

```
:help registers

:help copy-move
```

Register Help

```
:help i_CTRL-R
:help i_CTRL-R_CTRL-R
:help i_CTRL-R_CTRL-O
:help i_CTRL-R_CTRL-P
```

Ranges

- Line numbers (absolute or symbolic)
- Marks (manual or automatic)
- Search patterns (forward or backward)

Ranges

```
:help cmdline-ranges
```

Abbreviations

- **`iabbrev`** for Insert Mode
- **`cabbrev`** for Command-Line Mode
- **`abbrev`** for both
- Best for content expansion (vs. behavior)

Abbreviations

```
iabbrev teh      the
iabbrev tihs     this
iabbrev jsut     just
iabbrev exmaple example
```

Abbreviations

```
iabbrev _ett  wnodom@tmtowtdi.com
iabbrev _egm  billodom@gmail.com
iabbrev _sig  Thanks,<CR>Bill
```

Abbreviations

```
iabbrev <expr> _lipsum ''
\ . 'Lorem ipsum dolor sit amet, consectetur adipiscing elit. '
\ . 'Praesent hendrerit tempus adipiscing. Nullam vitae nibh ut '
\ . 'arcu volutpat porta. Ut tristique molestie mollis. In at erat '
\ . 'a ante molestie molestie. Suspendisse varius arcu nec augue '
\ . 'condimentum semper. Vestibulum malesuada pulvinar mi, '
\ . 'fermentum elementum eros adipiscing dignissim. Vestibulum at '
\ . 'odio risus. Ut ipsum erat, volutpat sed cursus nec, blandit '
\ . 'ac diam. Integer tincidunt sollicitudin eros, sit amet '
\ . 'vulputate justo sollicitudin malesuada. Nulla dapibus dictum '
\ . 'dolor id semper. Phasellus pretium dapibus massa quis auctor. '
```

Abbreviations

```
" Replace :Set typo with :set
" (See VimTip1285 for details.)
"
cabbrev Set <C-R>=(getcmdtype()==':'
\ && getcmdpos()==1 ? 'set' : 'Set')<CR>
```

Macros

The Basics

The Basics

- Record and repeat a series of keystrokes

The Basics

- Record and repeat a series of keystrokes
- Assigned to registers **a** through **z**

The Basics

The Basics

- Start recording with **`q<register>`**

The Basics

- Start recording with **`q<register>`**
- Do stuff

The Basics

- Start recording with **`q<register>`**
- Do stuff
- Stop recording with **`q`**

The Basics

The Basics

- Play back with **`@<register>`**

The Basics

- Play back with **`@<register>`**
- ...or **`<C-O>@<register>`** if you're already in Insert mode

A Simple Example

qwyyppv$r-kk.jq

A Simple Example (Really!)

A Simple Example (Really!)

`qw` - Start recording into register **`w`**

A Simple Example (Really!)

`qw` - Start recording into register **`w`**

`yy` - Yank (copy) current line

A Simple Example (Really!)

`qw` - Start recording into register **`w`**

`yy` - Yank (copy) current line

`pp` - Paste the copied line twice below current line

A Simple Example (Really!)

`qw` - Start recording into register **`w`**

`yy` - Yank (copy) current line

`pp` - Paste the copied line twice below current line

`v$` - Visually highlight to the end of the line

A Simple Example (Really!)

`qw` - Start recording into register **`w`**

`yy` - Yank (copy) current line

`pp` - Paste the copied line twice below current line

`v$` - Visually highlight to the end of the line

`r-` - Replace highlighted text with dashes

A Simple Example (Really!)

`qw` - Start recording into register **`w`**

`yy` - Yank (copy) current line

`pp` - Paste the copied line twice below current line

`v$` - Visually highlight to the end of the line

`r-` - Replace highlighted text with dashes

`kk` - Move down two lines

A Simple Example (Really!)

`qw` - Start recording into register **`w`**

`yy` - Yank (copy) current line

`pp` - Paste the copied line twice below current line

`v$` - Visually highlight to the end of the line

`r-` - Replace highlighted text with dashes

`kk` - Move down two lines

`.` - Repeat last edit

A Simple Example (Really!)

`qw` - Start recording into register **`w`**

`yy` - Yank (copy) current line

`pp` - Paste the copied line twice below current line

`v$` - Visually highlight to the end of the line

`r-` - Replace highlighted text with dashes

`kk` - Move down two lines

`.` - Repeat last edit

`j` - Move down one line

A Simple Example (Really!)

`qw` - Start recording into register **`w`**

`yy` - Yank (copy) current line

`pp` - Paste the copied line twice below current line

`v$` - Visually highlight to the end of the line

`r-` - Replace highlighted text with dashes

`kk` - Move down two lines

`.` - Repeat last edit

`j` - Move down one line

`q` - Stop recording

Macros == Registers

- View with **`:registers`** (or just **`:reg`**)
- View specific macros with **`:reg abc`**

Clearing a Macro

Clearing a Macro

- Assign an empty string:

```
:let @w = ''
```

Clearing a Macro

- Assign an empty string:

```
:let @w = ''
```

- ...or record nothing:

```
qwq
```

Duplicating a Macro

- Just use an assignment:

```
:let @z = @a
```

Combining Macros

Combining Macros

- `qz@a@b@cq`

Combining Macros

- `qz@a@b@cq`
- `:let @z = @a . @b . @c`

Combining Macros

- `qz@a@b@cq`
- `:let @z = @a . @b . @c`
- `:let @z = '@a@b@c'`

Editing a Macro

- Start an assignment statement:

  `:let @w = '`

- Paste the current value of the macro:

  `<C-R><C-R>w`

- Edit the contents
- Add the closing quote and press Return

Recursive Macros

- Just a macro that calls itself:
  `qwqqwgU$j@wq`
- Don't forget to empty the macro before recording!
- A little too easy to lock up Vim
- Usually easier to use a non-recursive macro with a large count

Using Counts with Macros

- `100@w`

Using Ranges with Macros

- `:5,50normal @w`
- `:'<,'>normal @w`

Mondo Macros

Mondo Macros

- `ql:set tw=33^M:1^MvGgq:set tw=44^M:1^MvG:centerq`

Mondo Macros

- `ql:set tw=33^M:1^MvGgq:set tw=44^M:1^MvG:centerq`
- `:bufdo normal @l`

Appending to a Macro

- `qW...q`

Macros or Maps?

- Macros: Great for one-shot tasks
- Keymaps: Better for more permanent customization

Free Macros!

- The Magical Dot Command
- `@:` - Repeat most recent Ex command
- `@@` - Repeat most recent macro

Resources

- `:help complex-repeat`
- `:help recording`
- `:help registers`

Key Mapping

What?

What?

- Bind a series of keystrokes to a key sequence

What?

- Bind a series of keystrokes to a key sequence
- Better for defining behavior (vs. abbreviations)

What?

- Bind a series of keystrokes to a key sequence
- Better for defining behavior (vs. abbreviations)
- Better for long-term customization (vs. macros)

What?

- Bind a series of keystrokes to a key sequence
- Better for defining behavior (vs. abbreviations)
- Better for long-term customization (vs. macros)
- Can be defined across all of Vim's modes

What?

- Bind a series of keystrokes to a key sequence
- Better for defining behavior (vs. abbreviations)
- Better for long-term customization (vs. macros)
- Can be defined across all of Vim's modes
- ***Fundamental to customizing Vim***

Map Command Structure

{cmd}	{attr}	{lhs}	{rhs}
map command	attributes	left-hand side	right-hand side

Map Command Structure

```
" Example map command
map <silent> <F2> :echo 'Hello!'<CR>
```

Map Commands

`omap`	`nmap`
`cmap`	`imap`
`lmap`	`vmap`
`map`	`xmap`
`map!`	`smap`

Map Commands

```
nnoremap
inoremap
vnoremap
xnoremap
snoremap
onoremap
cnoremap
```

Map Commands

```
nnoremap
inoremap
vnoremap
xnoremap
snoremap
onoremap
cnoremap
```

Map Attributes

Map Attributes

```
<silent>
<buffer>
<expr>
<unique>
<script>
<special>
```

Available Keys

- Function keys and shifted function keys
- **`<Leader>`** and **`<LocalLeader>`** sequences (especially in plugins and scripts)

Sorta-Available Keys

- Alt / Option / Meta Keys (safest in the GUI)
- Commands you don’t use
- Command “synonyms” you don’t use
- Comma
- Semicolon
- Underscore

Finding Keys to Map

`:help map-which-keys`

`:help index.txt`

`:help` *{key}*`<ctrl-d>`

Displaying Maps

`:nmap`

`:nmap` *{key}*

`:mkvimrc ~/scratch.vim`

Removing Maps

Removing Maps

:unmap *{key}*

Removing Maps

`:unmap` *{key}*

`:verbose map` *{key}*

Sample Maps

```
" Overload Control+L to also clear
" search highlighting
"
nnoremap <C-L>  :nohlsearch<CR><C-L>
```

Sample Maps

```
" Toggle screen wrapping of long lines,
" then display value of ‘wrap’ option
"
nnoremap <Leader>w
\   :set invwrap<Bar>set wrap?<CR>
```

Sample Maps

```
" Make up/down arrows move by display
" lines instead of physical lines
"

nnoremap <Up>      gk
nnoremap <Down>    gj
```

Sample Maps

```
" Reformat current paragraph
"
nnoremap Q  gqap
```

Sample Maps

```
"  Indent/unindent highlighted block
"  (and maintain highlight)
"
vnoremap <Tab>     >gv
vnoremap <S-Tab>   <gv
```

Sample Maps

```
" Disable paste-on-middle-click
"
inoremap <MiddleMouse>  <Nop>
```

Sample Maps

```
" Center display line after searches
"
nnoremap n    nzz
nnoremap N    Nzz
nnoremap *    *zz
nnoremap #    #zz
nnoremap g*   g*zz
nnoremap g#   g#z
```

Sample Maps

```
" Edit vimrc in new tab
"
nnoremap ,ev  :tabedit $MYVIMRC<CR>
```

Sample Maps

```
" Delete to end of line, bash-style
"
inoremap <C-K> <C-O>D
```

Sample Maps

```
" Control+Up/Down move lines & selections up and down.
" (Based on http://vim.wikia.com/wiki/VimTip646)
"
" Define maps for Normal and Visual modes, then re-use
" them for Insert and Select modes.
"
nnoremap <silent>  <C-Up>     :move -2<CR>
nnoremap <silent>  <C-Down>   :move +<CR>
xnoremap <silent>  <C-Up>     :move '<-2<CR>gv
xnoremap <silent>  <C-Down>   :move '>+<CR>gv
imap     <silent>  <C-Up>    <C-O><C-Up>
imap     <silent>  <C-Down>  <C-O><C-Down>
smap     <silent>  <C-Up>    <C-G><C-Up><C-G>
smap     <silent>  <C-Down>  <C-G><C-Down><C-G>
```

$VIMRUNTIME/mswin.vim

```
:vmap <Tab>     >gv
:vmap <S-Tab>   <gv
```

```
:nnoremap <F7>     gqap
:nnoremap <S-F7>   vipJ
```

Vim as Language

Caps Lo
Q
W
M
A
Z
Y
Tab
Backspace
Enter
Ctrl
Prt Sc
SysRq
Ins
F9
F10
F4
Home
End
Alt
Alt
H
U
N
#
3

Q
W
E
A
S
D
Z
X
C
$
4
%
5
^
6
&
7
R
T
Y
U
F
G
H
J
ctrl
alt

Verbs:

Operators & Commands

d

Subjects:

Motions & Text Objects

dj

Adjectives:

Ranges & Repetition

dioj

Objects:

Registers

"Ok.

Examples

0y$

yy

gg=G

"adG

gqap

vipJ

[No Name] + – VIM1

[No Name] [+]

```
 1
 2 Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Sed eu lorem s>
 3
 4 Fusce placerat dapibus purus. Proin id pede eu justo malesuada vehicula.>
 5
 6 Quisque dapibus turpis ut elit. In sem. Pellentesque eget arcu nec ipsum>
 7
 8 Quisque tincidunt. In hac habitasse platea dictumst. Aenean porta congue>
 9
10 Morbi congue. Sed at enim. Morbi ac turpis sit amet nulla pretium ornare>
11
12 Etiam sodales. Morbi ac lacus. Aenean facilisis eleifend erat. Ut ac ero>
13
14 Nulla adipiscing justo in ligula. Maecenas sollicitudin ligula ultricies>
15
16 Aliquam nulla felis, lobortis vitae, iaculis eget, molestie eu, nibh. Cu>
17
18 Cras sapien lorem, posuere volutpat, posuere ut, aliquet eget, elit. Cla>
19
20 Donec a dolor quis ipsum cursus interdum. In tincidunt vehicula lectus. >
21
~
[No Name] [+]                                                  2,1          All
"~/Work/vim-sandbox/samples/sample-text-unwrapped.txt" 20L, 6714C
```

[No Name] + – VIM1

[No Name] [+]

```
 1
 2 Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Sed eu
 3 lorem suscipit ipsum accumsan molestie. Phasellus iaculis enim
 4 vel sem. Maecenas vel nulla in justo fringilla mollis. Cras
 5 congue ullamcorper quam. Morbi ac massa. Vestibulum interdum eros
 6 nec sem auctor molestie. Praesent urna sem, lobortis et, rutrum
 7 feugiat, laoreet at, odio. Etiam elementum orci sed leo.
 8 Suspendisse ac elit. Phasellus lorem turpis, tempus at, aliquet
 9 sed, semper nec, justo. Cras et enim sit amet mauris tincidunt
10 luctus. Aliquam vulputate. Curabitur fringilla. In quis purus.
11
12 Fusce placerat dapibus purus. Proin id pede eu justo malesuada vehicula.>
13
14 Quisque dapibus turpis ut elit. In sem. Pellentesque eget arcu nec ipsum>
15
16 Quisque tincidunt. In hac habitasse platea dictumst. Aenean porta congue>
17
18 Morbi congue. Sed at enim. Morbi ac turpis sit amet nulla pretium ornare>
19
20 Etiam sodales. Morbi ac lacus. Aenean facilisis eleifend erat. Ut ac ero>
21
22 Nulla adipiscing justo in ligula. Maecenas sollicitudin ligula ultricies>
[No Name] [+]                                          1,0-1          Top
```

vat

“ayis

kyyp^v$r=ja

[No Name] + – VIM1

[No Name] [+]

```
1
2 Header
3
4 Subheader 1
5
6 Plain text.
7
8
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
[No Name] [+]                                   2,1           All
```

[No Name] + – VIM1

[No Name] [+]

```
1
2 Header
3 ======
4
5 Subheader 1
6 -----------
7
8 Plain text.
9
10
~
~
~
~
~
~
~
~
~
~
~
~
~
[No Name] [+]                                    1,0-1          All
```

Files
and **Buffers**
and **Windows**
and **Tabs**
and **Instances**

Definitions:

File

- Um... a file

Buffer

- Usually associated with a file, but doesn't have to be
- Usually has a name, but doesn't have to
- Doesn't have to be visible
- Several different types
  (that you mostly don't have to care about)

Window

- Displays the contents of a buffer
- Always at least one window open
- More than one window can display the same buffer

Tab

- A group of windows
- Not like tabs in other editors
- Sometimes called “tab pages”
- Better name might be “layout” or “viewport”

Buffers, Windows, Tab Pages

- Often confused for one another
- A buffer is the contents of a (potential) file
- A window is a place where a buffer can be viewed
- A tab page is a collection of windows
- A buffer can be viewed in any number of windows

Instance

- A single running copy of Vim
- GUI or terminal
- Doesn’t share (well) with other Vim instances

Loading Multiple Files

- `vim *.sh`
- `vim foo.txt bar.txt baz.txt`
- `vim -o widget.c header.h`
- `vim -p widget.c header.h`

Loading Multiple Files

- **`:edit foo.txt`**
- **`:args *.sh`**
- **`:args foo.txt bar.txt baz.txt`**
- The args list is modifiable with **`:argadd`** and **`:argdel`**

Moving Among Buffers

- **`:set hidden`**
- **`:ls`**
- **`:b`** *N*
- **`:b`** *name*
- **`:bfirst`** **`:blast`** **`:bnext`** **`:bprev`**
- **`<C-^>`**

Closing Buffers

- **`:bdelete`**
- **`:bd`** *N*
- **`:bd`** *name*
- **`:bd!`**

Buffer Settings

- **hidden**
- **switchbuf**

Buffer Commands

- `:ball`
- `:bufdo`

Creating Windows

- `:split`
- `:vsplit`
- `<C-W> s`
- `<C-W> v`

Moving Between Windows

- **`<C-W> h`**
- **`<C-W> j`**
- **`<C-W> k`**
- **`<C-W> l`**

Moving Between Windows

- **<C-W> p**
- **<C-W> <C-W>**

Resizing Windows

- **`<C-W> +`**
- **`<C-W> -`**
- **`<C-W> <`**
- **`<C-W> >`**
- ...or just use the mouse

Resizing Windows

- **`<C-W> =`**
- **`<C-W> _`**
- **`<C-W> |`**

Moving Windows

- **`<C-W> H`**
- **`<C-W> J`**
- **`<C-W> K`**
- **`<C-W> L`**

- **`<C-W> x`**

Closing Windows

- `:q`
- `<C-W> c`
- `:hide`

Closing *Other* Windows

- `:only`
- `<C-W> o`

Window Commands

- `:vertical`
- `:windo`
- `:wincmd`

Window Keys

- More than you'll ever need (or want)
- Good candidates for your own key maps
- **`:help windows.txt`**
- **`:help ctrl-w`**

Window Settings

- **`splitbelow`**
- **`splitright`**
- **`winminheight`**
- **`winminwidth`**
- **`winheight`**

Creating Tab Pages

- `:tabnew`
- `:tabedit`
- `<C-W> T`

Moving Between Tab Pages

- **`gt`**
- **`gT`**
- ***N*`gt`**
- Often mapped to OS-specific keystrokes

Moving Tab Pages

- **`:tabmove`**
- **`:tabmove`** *N*
- ...or just use the mouse (in the GUI)

Closing Tab Pages

- `:q` in last window
- `:tabclose`
- GUI often provides a close button

Closing *Other* Tab Pages

- `:tabclose` *N*
- `:tabonly`

Tab Page Settings

- **`tabline`**
- **`guitablabel`**
- May be OS-specific options as well

Tab Page Commands

- `:tab`
- `:tabs`
- `:tabdo`

What Good are Tab Pages?

- Full-page help, docs, etc.
- Alternate layouts
- Temporary work
- Diffs
- Multiple projects
- Mini-sessions

Instances

- A single instance includes all buffers, windows, tabs, settings, etc.
- Easy to create more than one, especially in the terminal
- ...but that's usually not necessary
- ...and usually isn't a good idea

Configuration & Customizaton

The Usual Suspects

- `.vimrc`
- `.gvimrc`
- `~/.vim/`

The minimal .vimrc

The minimal .vimrc

this slide intentionally left blank

Strict vi Compatibility?

Strict vi Compatibility?

```
:set nocompatible
```

Extensions

```
:filetype plugin indent on
```

Backups and Swapfiles

```
:set backupdir-=.
:set backupdir^=~/tmp,/tmp
:set directory=~/tmp//,.
```

Backups and Swapfiles

```
:set backupdir-=.
:set backupdir^=$TEMP
:set directory=$TEMP\\\\
```

Easier Searching

Easier Searching

```
:set incsearch
:set hlsearch
:set ignorecase
:set smartcase
```

Controlling Whitespace

Controlling Whitespace

```
:set tabstop=4
:set softtabstop=4
:set expandtab
:set smarttab
```

Controlling Whitespace

```
:set shiftwidth
:set shiftround
```

Roam If You Want To

```
:set backspace=indent,eol,start
:set whichwrap+=<,>,[,]
```

Better Buffers

```
:set hidden
```

Blessed Silence

```
:set vb
:set t_vb=
```

Exploring ~/.vim

- Colorschemes
- Extra filetypes
- Custom syntax and indentation
- Bundles
- And more

Search and Substitution

Regexes - Feel the Power

- Regular expressions (regexes) are the patterns used in searches
- Formal language to concisely describe strings
- Vim's regexes are among the most powerful available

What They're Not

- Filename "globs" (but there's a little overlap)
- All the same everywhere (but there are lots of similarities among the flavors)
- Line noise (but you might be forgiven for thinking so)

Literal Characters

- Match themselves (unless they're escaped)
- Any alphanumeric character
- Whitespace
- Punctuation other than `.` `[` `]` `\` `^` `$`

Special Characters

- For modifiers, anchors, groups, character classes, etc.
- The punctuation we just saw: `.` `[` `]` `\` `^` `$`
- Nothing else special unless it's escaped

"Escaped"?

- Adding a backslash before a special character converts it to a literal (so you can match, say, a period)
- Adding a backslash before a literal character often converts it to a special character (like **`\n`** or **`\t`**)

Anchors

- "Anchor" expression to start of a line with a caret: **^**
- ...or to the end of a line with the dollar sign: **$**
- Match "word boundaries" with **\<** and **\>**

Repetition

- Match zero or more of something with asterisk: `*`
- One or more with (escaped!) plus: `\+`
- Exactly *n*: `\{`*n*`}`
- At least *n*: `\{`*n*`,}`
- No more than *m*: `\{,`*m*`}`
- At least n but no more than *m*: `\{`*n*`,`*m*`}`

Character Classes

- Unescaped square brackets
- Any character: **`[abc]`**
- Or a range of characters: **`[a-f]`**
- Any character not in the class: **`[^a-f]`**
- Collections: **`[[:alpha:]]`** **`[[:print:]]`** etc. (note the doubled brackets)

The Dot

- An unescaped period matches any character
- Well, except a newline
- And except when it's in a character class: `[.]`

Alternation

- Match *this* or *that*: **`this\|that`**
- Note the escape!

Grouping and Capturing

- Use escaped parentheses: **`\(pattern\)`**
- Refer to the captured group with backrefs: **`\1  \2`**

Um... this just keeps going

- Yes.
- Yes it does.
- How about just **`:help pattern.txt`**
- Oh, and a possibly crazy person's online tip archive

Plugins

Plugins to Consider

- Command-T, CtrlP, etc.
- NERD tree
- Ack
- JSHint
- Fugitive
- Matchit
- Surround
- Tabular
- NERD commenter
- Easymotion
- Syntastic
- Taglist
- BufExplorer / MiniBufExpl
- Many many (many) more

Managing Plugins

- Manual
- Pathogen
- Vundle
- A few others, some old, some new

A Little Scripting Interlude

Resources

- **21st Century Vim Slides**
  http://www.billodom.com/vim/21st-century-vim-strange-loop-2013.pdf
- **Vim**
  http://www.vim.org/
- **MacVim**
  http://code.google.com/p/macvim/
- **Vim Plugins, Add-Ons, Scripts**
  http://www.vim.org/scripts/index.php
- **Vim Mailing Lists**
  http://www.vim.org/maillist.php

Resources

- **Vimcasts**
  http://vimcasts.org/
- **Derek Wyatt's Vim Tutorial Videos**
  http://www.derekwyatt.org/vim/vim-tutorial-videos/

Resources

- **godlygeek's explanation of buffers, windows, and tabs**
  http://billodom.com/vim/vim-buffers-windows-tabs-explanation.txt
- **VimCasts - How to use buffers/windows/tabs**
  http://vimcasts.org/episodes/working-with-buffers/
  http://vimcasts.org/episodes/working-with-windows/
  http://vimcasts.org/episodes/working-with-tabs/
  http://vimcasts.org/episodes/how-to-use-tabs/
- **Derek Wyatt - Working with Many Files**
  http://vimeo.com/6306508
- **Vim Tips Wiki - Tabs**
  http://vim.wikia.com/wiki/Category:Tabs

Resources

- **Efficient Editing With Vim**
  http://jmcpherson.org/editing.html
- **Zenclavier: Extreme Keyboarding**
  http://www.oreilly.com/news/zenclavier_1299.html
- **Your problem with Vim is that you don't grok vi**
  http://stackoverflow.com/questions/1218390/#1220118
- **Everyone Who Tried to Convince Me to use Vim was Wrong**
  http://yehudakatz.com/2010/07/29/everyone-who-tried-to-convince-me-to-use-vim-was-wrong/
- **Damian Conway's "Scripting the Vim Editor" Series**
  https://www.ibm.com/developerworks/linux/library/l-vim-script-1/

Resources

- **Bill Odom's Vim Environment**
  https://github.com/wnodom/wnodom-vim-environment
- **Steven Pritchard's Vim Environment**
  https://github.com/silug/vim-environment
- **Damian Conway's Vim Environment**
  https://github.com/thoughtstream/Damian-Conway-s-Vim-Setup
- **Steve Losh's Vim Environment**
  https://github.com/sjl
- **Tim Pope's Vim Environment**
  https://github.com/tpope/tpope

Resources

- **Practical Vim *by* Drew Neil**
  http://pragprog.com/book/dnvim/practical-vim/
- **Learning the vi and Vim Editors *by* Arnold Robbins, et al**
  http://shop.oreilly.com/product/9780596529833.do
- **Learn Vimscript the Hard Way *by* Steve Losh**
  http://learnvimscriptthehardway.stevelosh.com
- **Vi iMproved (VIM) *by* Steve Oualline**
  http://www.truth.sk/vim/vimbook-OPL.pdf

Resources

- **Vim Geeks**
  http://vimgeeks.org/
- **@vimgeeks**
  http://twitter.com/vimgeeks
- **Vim-Related Twitter Accounts**
  https://twitter.com/#!/VimGeeks/following

Thanks!