Pasting code with syntax coloring in emails

<http://vim.wikia.com/wiki/Pasting_code_with_syntax_coloring_in_emails>

## Creating html[Edit](http://vim.wikia.com/wiki/Pasting_code_with_syntax_coloring_in_emails?action=edit&section=1)

In gvim, use "Convert to HTML" on the Syntax menu. A new file is created, containing the html equivalent of all text from the current buffer.

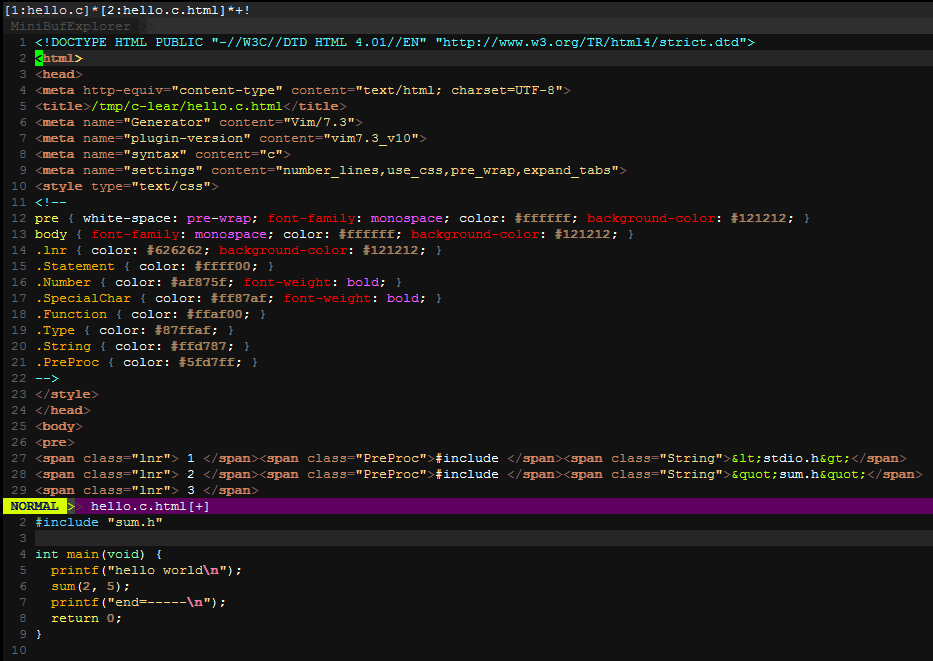
The same operation can be performed by entering a command:

:TOhtml

If you visually select some lines before entering :TOhtml, only the selected lines will be converted.

If your color scheme uses a dark background, you may want to temporarily switch to a white background before creating the HTML document. Copy-paste of rich text from the browser into an email client or other application often discards the background color. One way to do that is to use the command:

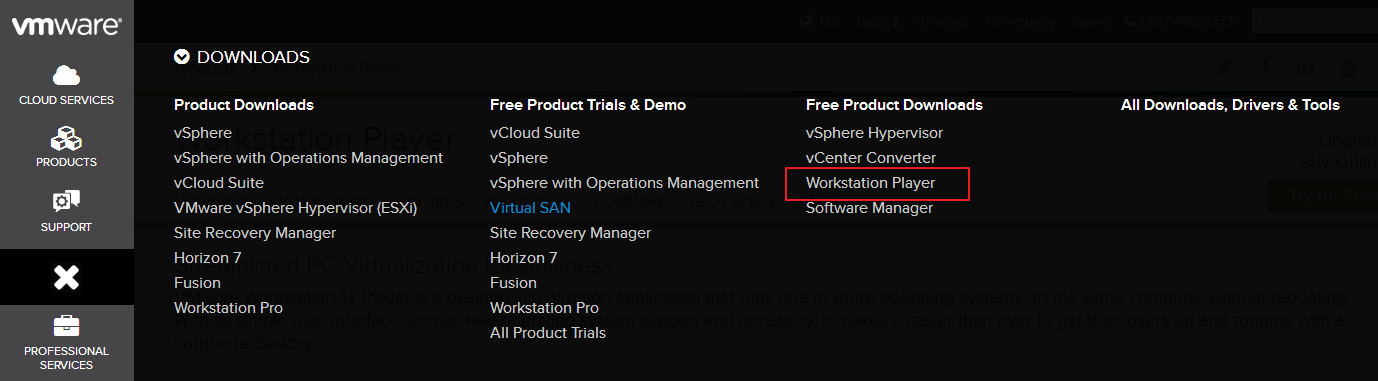
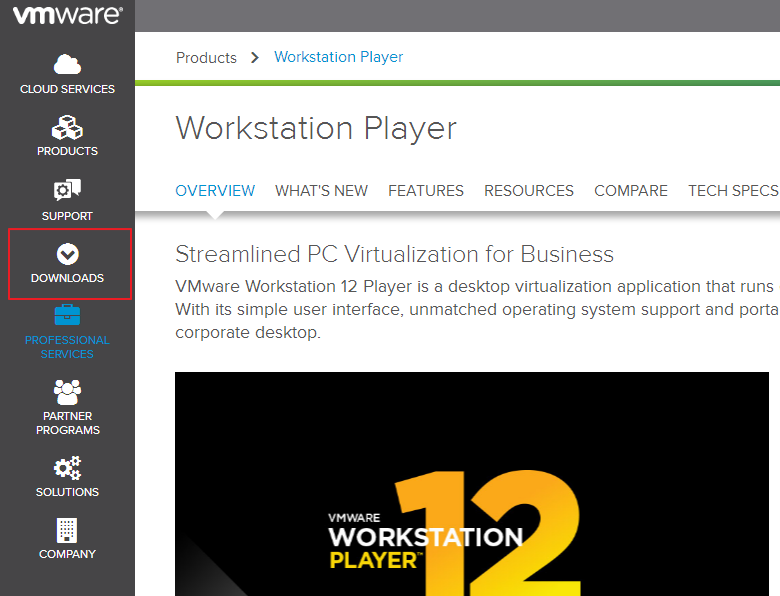
:colorscheme default



<http://www.pendrivelinux.com/yumi-multiboot-usb-creator/>

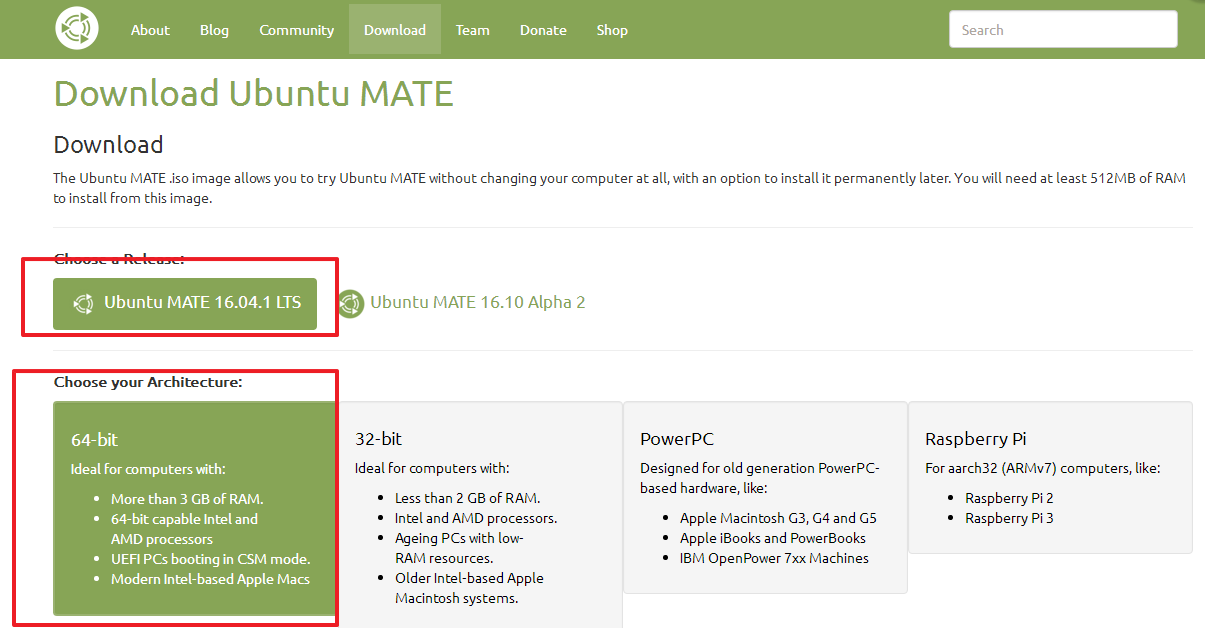


<http://www.vmware.com/products/player.html>



<https://ubuntu-mate.org/>

<https://ubuntu-mate.org/download/>



# ssh（安全外壳协议）

SSH 为 [Secure Shell](http://baike.baidu.com/view/2118359.htm) 的缩写，由 IETF 的网络小组（Network Working Group）所制定；SSH 为建立在应用层和传输层基础上的安全协议。SSH 是目前较可靠，专为[远程登录](http://baike.baidu.com/view/59099.htm)会话和其他网络服务提供安全性的协议。利用 SSH 协议可以有效防止远程管理过程中的信息泄露问题。SSH最初是UNIX系统上的一个程序，后来又迅速扩展到其他操作平台。SSH在正确使用时可弥补网络中的漏洞。SSH客户端适用于多种平台。几乎所有UNIX平台—包括[HP-UX](http://baike.baidu.com/view/58963.htm)、[Linux](http://baike.baidu.com/view/1634.htm)、[AIX](http://baike.baidu.com/view/349664.htm)、[Solaris](http://baike.baidu.com/subview/329359/5113665.htm)、[Digital](http://baike.baidu.com/view/428214.htm) [UNIX](http://baike.baidu.com/view/8095.htm)、[Irix](http://baike.baidu.com/view/3373083.htm)，以及其他平台，都可运行SSH。

OpenSSH

OpenSSH 是 [SSH](http://baike.baidu.com/view/16184.htm) （Secure [SHell](http://baike.baidu.com/view/849.htm)） 协议的免费开源实现。SSH协议族可以用来进行远程控制， 或在计算机之间传送文件。而实现此功能的传统方式，如telnet([终端仿真](http://baike.baidu.com/view/190233.htm)协议)、 rcp ftp、 rlogin、rsh都是极为不安全的，并且会使用明文传送密码。OpenSSH提供了服务端后台程序和客户端工具，用来加密远程控件和文件传输过程中的数据，并由此来代替原来的类似服务。

OpenSSH是使用SSH透过计算机网络加密通讯的实现。它是取代由SSH Communications Security所提供的商用版本的开放源代码方案。目前OpenSSH是OpenBSD的子计划。

OpenSSH常常被误认以为与OpenSSL有关联，但实际上这两个计划的有不同的目的，不同的发展团队，名称相近只是因为两者有同样的软件发展目标──提供开放源代码的加密通讯软件。

installl openssh-server

<https://help.ubuntu.com/community/SSH/OpenSSH/Configuring>

sudo apt-get install openssh-server

# Ubuntu Linux: Start / Stop / Restart OpenSSH ( SSH ) Server

<http://www.cyberciti.biz/faq/howto-start-stop-ssh-server/>

$ sudo /etc/init.d/ssh start

$ sudo /etc/init.d/ssh stop

$ sudo /etc/init.d/ssh restart

$ sudo /etc/init.d/ssh status

设置阿里镜像源

<http://mirrors.aliyun.com/>

<http://mirrors.aliyun.com/help/ubuntu>

(注意：此处只有成功了才能成功执行sudo apt-get install screen等这样的安装操作)

apt-get是一条[linux命令](http://baike.baidu.com/view/97023.htm)，适用于deb包管理式的操作系统，主要用于自动从互联网的软件仓库中搜索、安装、升级、[卸载](http://baike.baidu.com/view/386432.htm)软件或操作系统。

Advanced Package Tool，又名apt-get，是一款适用于[Unix](http://baike.baidu.com/view/8095.htm)和[Linux](http://baike.baidu.com/view/1634.htm)系统的应用程序管理器。

“dpkg ”是“Debian Packager ”的简写。为 “Debian” 专门开发的套件管理系统，方便软件的安装、更新及移除。所有源自“Debian”的“Linux ”发行版都使用 “dpkg”，例如 “Ubuntu”、“Knoppix ”等。

Install screen tools

**sudo apt-get install screen**

screen的使用参考<https://www.digitalocean.com/community/tutorials/how-to-install-and-use-screen-on-an-ubuntu-cloud-server>

# How to Install and Use Screen on an Ubuntu Cloud Server

<http://unix.stackexchange.com/questions/1045/getting-256-colors-to-work-in-tmux>

# [Getting 256 colors to work in tmux](http://unix.stackexchange.com/questions/1045/getting-256-colors-to-work-in-tmux)

<http://stackoverflow.com/questions/6787734/strange-behavior-of-vim-color-inside-screen-with-256-colors>

# [Strange behavior of vim color inside screen with 256 colors](http://stackoverflow.com/questions/6787734/strange-behavior-of-vim-color-inside-screen-with-256-colors)

如下的配置可以解决vim在screen中 颜色不一致的问题。

.screenrc

hardstatus alwayslastline "%{.bW}%-w%{.rW}%n %t%{-}%+w %=%{..G} %H %{..Y} %Y-%m-%d %c"

shell "bash"

startup\_message off

vbell off

altscreen on

attrcolor b ".I"

defbce "on"

termcapinfo xterm\* 'is=\E[r\E[m\E[2J\E[H\E[?7h\E[?1;4;6l'

termcapinfo xterm 'Co#256:AB=\E[48;5;%dm:AF=\E[38;5;%dm'

term screen-256color

<http://superuser.com/questions/212392/how-to-include-screens-session-name-in-hardstatus>

# [how to include screen's session name in hardstatus?](http://superuser.com/questions/212392/how-to-include-screens-session-name-in-hardstatus)

You can include this string (with additional information, if desired) in your $PS1:

\[\e]0;$STY\007

inside single quotes in order to delay evaluation of the variable. Then add this to your ~/.screenrc:

hardstatus string '%H:%h'

Unfortunately, screen doesn't set $STY in the environment of the commands it spawns for the backtick feature.

Another option, instead of the one above:

hardstatus string '%H:%`'

backtick 0 30 30 sh -c 'screen -ls | grep --color=no -o "$PPID[^[:space:]]\*"'

The advantage of this one is that it follows changes made by using the sessionname command. The first option doesn't.

**Edit:**

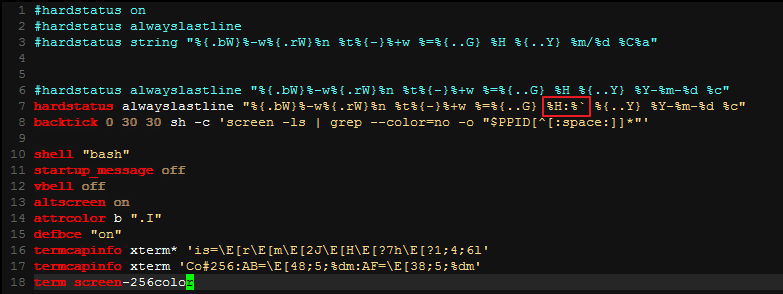
From [here](http://old.nabble.com/Re%3a-Session-name-in-status-line-p22456825.html):

Since $STY is not set yet when screen sources .screenrc, you can use this trick in your .screenrc:

screen

screen $SHELL -c 'screen -X caption always "$STY"'

I.e. send a screen command to the first window.



linux screen:

<http://aperiodic.net/screen/quick_reference>

change to window by number or name C-a ' <number or title>

rename current window C-a A

|  |  |
| --- | --- |
| create new window | [C-a c](http://aperiodic.net/screen/commands:screen) |

C-a n

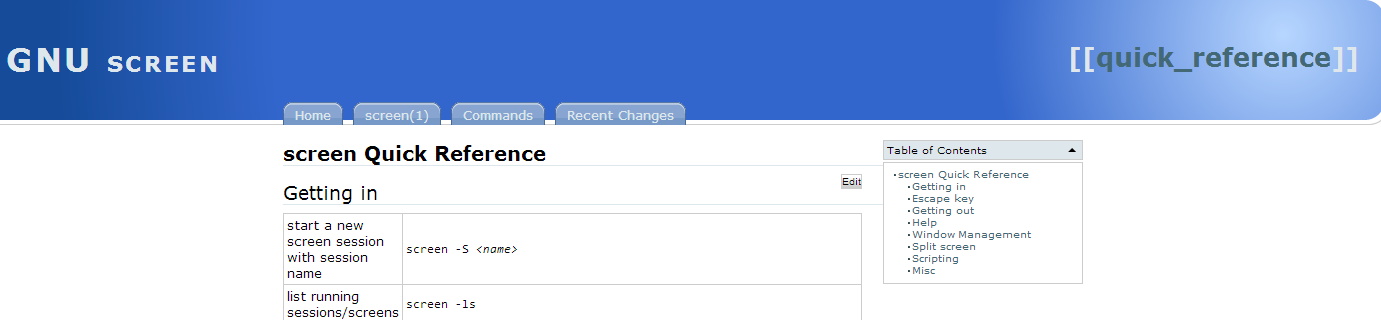
C-a p

C-a <Number>

scree –r …

screen –d …

screen -ls



<http://superuser.com/questions/458122/gnu-screen-kill-a-frozen-tab>

Gnu screen: kill a frozen tab

try to kill command.

select stalled window ( C-a :select <stalled window number> )

send kill command ( C-a :kill )

安装git

sudo apt-get install git



# [How to find out which versions of a package can I install on APT](http://superuser.com/questions/393681/how-to-find-out-which-versions-of-a-package-can-i-install-on-apt)

<http://superuser.com/questions/393681/how-to-find-out-which-versions-of-a-package-can-i-install-on-apt>

Just as an addendum

apt-cache madison <<package name>>

will list the versions available from all your sources.

apt-cache madison vim

vim | 2:7.3.547-1 | http://debian.mirrors.tds.net/debian/ unstable/main amd64 Packages

vim | 2:7.3.429-2 | http://debian.mirrors.tds.net/debian/ testing/main amd64 Packages

vim | 2:7.3.429-2 | http://http.us.debian.org/debian/ testing/main amd64 Packages

vim | 2:7.3.429-2 | http://debian.mirrors.tds.net/debian/ testing/main Sources

vim | 2:7.3.547-1 | http://debian.mirrors.tds.net/debian/ unstable/main Sources

# [How to check the version before install packages using apt-get](http://stackoverflow.com/questions/18885820/how-to-check-the-version-before-install-packages-using-apt-get)

<http://stackoverflow.com/questions/18885820/how-to-check-the-version-before-install-packages-using-apt-get>

OK, I found it.

apt-cache policy <package name> will show the version details.

It also shows which version is currently installed and which versions are available to install.

For example, apt-cache policy hylafax+

<http://www.cyberciti.biz/faq/debian-linux-install-gnu-gcc-compiler/>

<http://askubuntu.com/questions/154402/install-gcc-on-ubuntu-12-04-lts>

[**nixCraft**](http://www.cyberciti.biz/faq)

# Debian Linux Install GNU GCC Compiler and Development Environment

[a] **build-essential** package – Installs the following collection to compile c/c++ program on Debian/Ubuntu Linux:

1. libc6-dev – C standard library.
2. gcc – C compiler.
3. g++ – C++ compiler.
4. make – GNU make utility to maintain groups of programs.
5. dpkg-dev – Debian package development tools.

Basically, **build-essential** package contains an informational list of packages which are considered essential for building Debian packages including gcc compiler, make and other required tools. This package also depends on the packages on that list, to make it easy to have the build-essential packages installed.

$ sudo apt-get install build-essential

### Verify installation

You can verify gcc compiler and make tool using the following syntax:  
# whereis gcc make  
# gcc -v  
# make -v

源码安装vim （当前版本已经为vim8,而apt-get源中却还为7.4版本，所以选择源码安装）

<https://github.com/Valloric/YouCompleteMe/wiki/Building-Vim-from-source>

# [Valloric](https://github.com/Valloric)/[YouCompleteMe](https://github.com/Valloric/YouCompleteMe)

Compiling Vim from source is actually not that difficult. Here's what you should do:

1：First, install all the prerequisite libraries, including Git. For a Debian-like Linux distribution like Ubuntu, that would be the following:

sudo apt-get install libncurses5-dev libgnome2-dev libgnomeui-dev \

libgtk2.0-dev libatk1.0-dev libbonoboui2-dev \

libcairo2-dev libx11-dev libxpm-dev libxt-dev python-dev \

python3-dev ruby-dev git

(If you know what languages you'll be using, feel free to leave out packages you won't need, e.g. Python2 python-dev or Ruby ruby-dev. This principle heavily applies to the whole page.)

2: Remove vim if you have it already.

sudo apt-get remove vim vim-runtime gvim

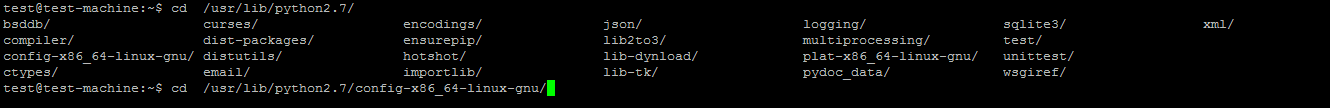
On Ubuntu 12.04.2 you probably have to remove these packages as well:

sudo apt-get remove vim-tiny vim-common vim-gui-common vim-nox

3: Once everything is installed, getting the source is easy.

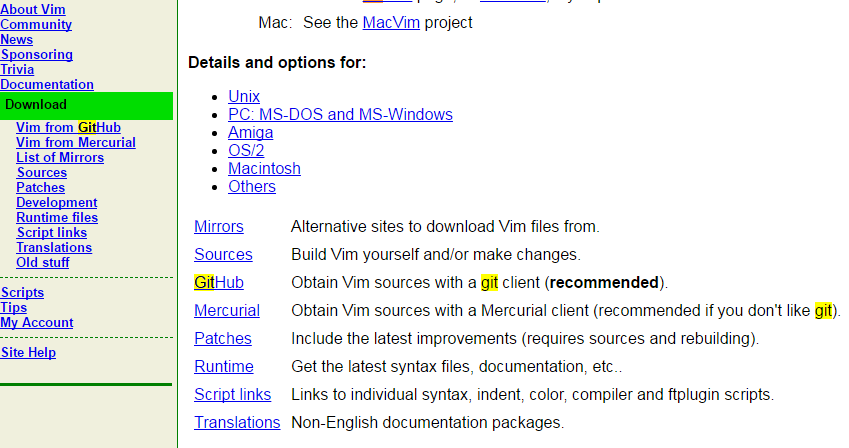
Note: If you are using Python, your config directory might have a machine-specific name (e.g.config-3.5m-x86\_64-linux-gnu). Check in /usr/lib/python[2/3/3.5] to find yours, and change the python-config-dir and/or python3-config-dir arguments accordingly.

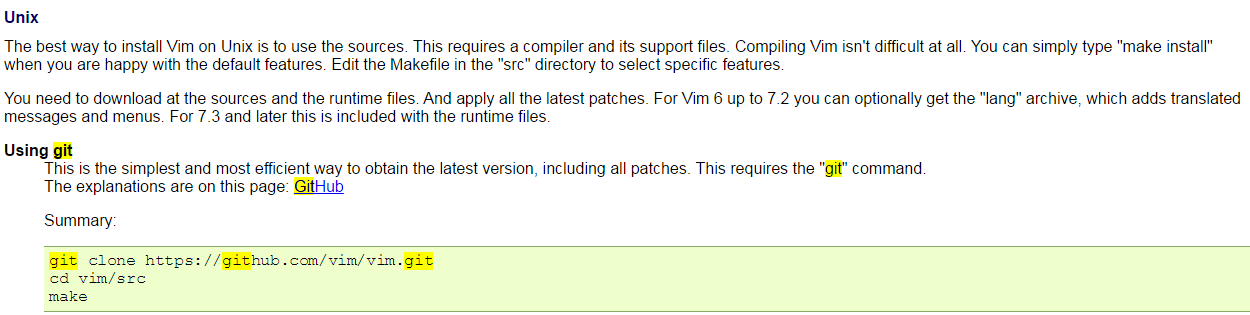
Also, if you're not using vim 7.4, make sure to set the VIMRUNTIMEDIR variable correctly below (for instance, with vim 7.4a, use /usr/share/vim/vim74a). Keep in mind that some vim installations are located directly inside /usr/share/vim; adjust to fit your system:





<http://www.vim.org/download.php>





cd /usr/local/src

sudo git clone https://github.com/vim/vim.git

cd vim

sudo ./configure --with-features=huge \

--enable-multibyte \

--enable-rubyinterp \

--enable-pythoninterp \

--with-python-config-dir=/usr/lib/python2.7/config-x86\_64-linux-gnu \

--enable-python3interp \

--with-python3-config-dir=/usr/lib/python3.5/config-3.5m-x86\_64-linux-gnu \

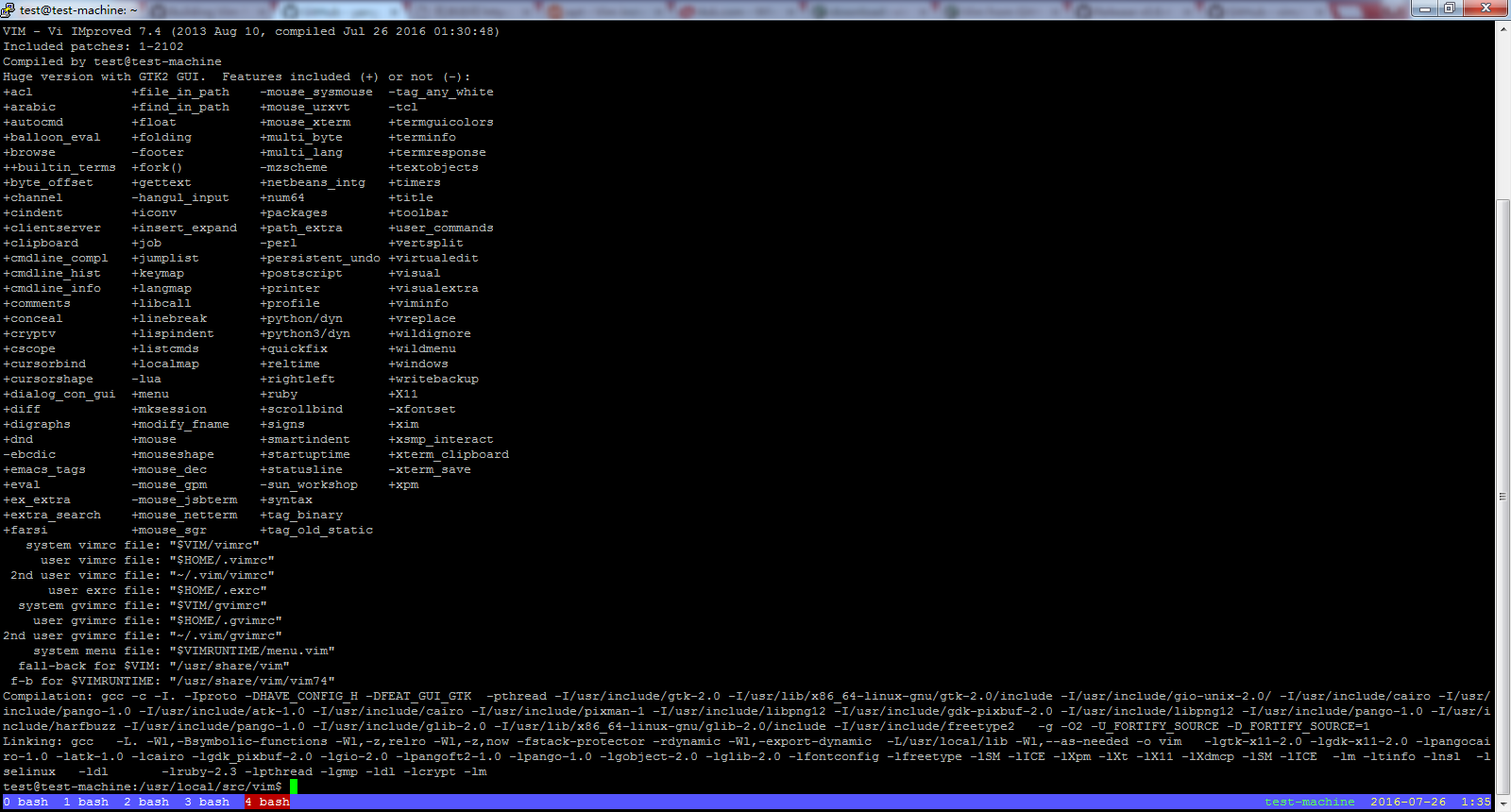
--enable-perlinterp \

--enable-luainterp \

--enable-gui=gtk2 --enable-cscope --prefix=/usr

sudo make VIMRUNTIMEDIR=/usr/share/vim/vim74

sudo make install



# 安装[VundleVim](https://github.com/VundleVim)/[Vundle.vim](https://github.com/VundleVim/Vundle.vim)**插件**

Set up [Vundle](http://github.com/VundleVim/Vundle.vim):

$ git clone https://github.com/VundleVim/Vundle.vim.git ~/.vim/bundle/Vundle.vim

Configure Plugins:：

Put this at the top of your .vimrc to use Vundle. Remove plugins you don't need, they are for illustration purposes.

set nocompatible " be iMproved, required

filetype off " required

" set the runtime path to include Vundle and initialize

set rtp+=~/.vim/bundle/Vundle.vim

call vundle#begin()

" alternatively, pass a path where Vundle should install plugins

"call vundle#begin('~/some/path/here')

" let Vundle manage Vundle, required

Plugin 'VundleVim/Vundle.vim'

" The following are examples of different formats supported.

" Keep Plugin commands between vundle#begin/end.

" plugin on GitHub repo

Plugin 'tpope/vim-fugitive'

" plugin from http://vim-scripts.org/vim/scripts.html

Plugin 'L9'

" Git plugin not hosted on GitHub

Plugin 'git://git.wincent.com/command-t.git'

" git repos on your local machine (i.e. when working on your own plugin)

Plugin 'file:///home/gmarik/path/to/plugin'

" The sparkup vim script is in a subdirectory of this repo called vim.

" Pass the path to set the runtimepath properly.

Plugin 'rstacruz/sparkup', {'rtp': 'vim/'}

" Install L9 and avoid a Naming conflict if you've already installed a

" different version somewhere else.

Plugin 'ascenator/L9', {'name': 'newL9'}

" All of your Plugins must be added before the following line

call vundle#end() " required

filetype plugin indent on " required

" To ignore plugin indent changes, instead use:

"filetype plugin on

"

" Brief help

" :PluginList - lists configured plugins

" :PluginInstall - installs plugins; append `!` to update or just :PluginUpdate

" :PluginSearch foo - searches for foo; append `!` to refresh local cache

" :PluginClean - confirms removal of unused plugins; append `!` to auto-approve removal

"

" see :h vundle for more details or wiki for FAQ

" Put your non-Plugin stuff after this line

Install Plugins:

Launch vim and run :PluginInstall

To install from command line: vim +PluginInstall +qall

# [viruschidai](https://github.com/viruschidai)/[vim-as-nodejs-ide](https://github.com/viruschidai/vim-as-nodejs-ide)

<https://github.com/viruschidai/vim-as-nodejs-ide/tree/master/vim-files/vim>

|  |
| --- |
| " Syntax hightlight " |
|  | syntax on |
|  |  |
|  | " Make vim incompatbile to vi. |
|  | set nocompatible |
|  | set modelines=0 |
|  |  |
|  | "TAB settings. |
|  | set tabstop=2 |
|  | set shiftwidth=2 |
|  | set softtabstop=2 |
|  | set expandtab |
|  |  |
|  | " More Common Settings. |
|  | set encoding=utf-8 |
|  | set scrolloff=3 |
|  | set autoindent |
|  | set showmode |
|  | set showcmd |
|  | set hidden |
|  | set wildmenu |
|  | set wildmode=list:longest |
|  | set visualbell |
|  |  |
|  | set cursorline |
|  | set ttyfast |
|  | set ruler |
|  | set backspace=indent,eol,start |
|  | set laststatus=2 |
|  |  |
|  | "set relativenumber |
|  | set number |
|  | set norelativenumber |
|  |  |
|  | "set undofile |
|  | set shell=/bin/bash |
|  | set lazyredraw |
|  | set matchtime=3 |
|  |  |
|  | "Changing Leader Key |
|  | let mapleader = "," |
|  |  |
|  | " Set title to window |
|  | set title |
|  | " Dictionary path, from which the words are being looked up. |
|  | set dictionary=/usr/share/dict/words |
|  |  |
|  | " Make pasting done without any indentation break." |
|  | set pastetoggle=<F3> |
|  |  |
|  | " Make Vim able to edit corntab fiels again. |
|  | set backupskip=/tmp/\*,/private/tmp/\*" |
|  |  |
|  | " Enable Mouse |
|  | set mouse=a |
|  |  |
|  | "Settings for Searching and Moving |
|  | nnoremap / /\v |
|  | vnoremap / /\v |
|  | set ignorecase |
|  | set smartcase |
|  | set gdefault |
|  | set incsearch |
|  | set showmatch |
|  | set hlsearch |
|  | nnoremap <leader><space> :noh<cr> |
|  | nnoremap <tab> % |
|  | vnoremap <tab> % |
|  |  |
|  |  |
|  | " Make Vim to handle long lines nicely. |
|  | set wrap |
|  | set textwidth=79 |
|  | set formatoptions=qrn1 |
|  |  |
|  | " Compile coffee script and goto specified line |
|  | autocmd FileType coffee,js,jade,styl,css,json autocmd BufWritePre <buffer> :%s/\s\+$//e |
|  | autocmd FileType coffee command! -nargs=1 C silent CoffeeCompile | :<args> |
|  |  |
|  | " Syntax highlight for github .md files " |
|  | au BufRead,BufNewFile \*.md set filetype=markdown |
|  |  |
|  | " pretty print json |
|  | nnoremap <C-S-F> :%!python -mjson.tool<CR> |
|  |  |
|  | "set colorcolumn=79 |
|  |  |
|  | " To show special characters in Vim |
|  | "set list |
|  | set listchars=tab:▸\ ,eol:¬ |
|  |  |
|  | " Naviagations using keys up/down/left/right |
|  | " Disabling default keys to learn the hjkl |
|  | nnoremap <up> <nop> |
|  | nnoremap <down> <nop> |
|  | nnoremap <left> <nop> |
|  | nnoremap <right> <nop> |
|  | "inoremap <up> <nop> |
|  | "inoremap <down> <nop> |
|  | "inoremap <left> <nop> |
|  | "inoremap <right> <nop> |
|  | nnoremap j gj |
|  | nnoremap k gk |
|  |  |
|  | " Rope settings." |
|  | inoremap <leader>j <ESC>:RopeGotoDefinition<cr> |
|  |  |
|  | " Get Rid of stupid Goddamned help keys |
|  | inoremap <F1> <ESC> |
|  | nnoremap <F1> <ESC> |
|  | vnoremap <F1> <ESC> |
|  |  |
|  | " Map : to ; also in command mode. |
|  | nnoremap ; : |
|  |  |
|  | " Set vim to save the file on focus out. |
|  | au FocusLost \* :wa |
|  |  |
|  | " Adding More Shorcuts keys using leader kye. |
|  | " Leader Kye provide separate namespace for specific commands. |
|  | ",W Command to remove white space from a file. |
|  | nnoremap <leader>W :%s/\s\+$//<cr>:let @/=''<CR> |
|  |  |
|  | " ,ft Fold tag, helpful for HTML editing. |
|  | nnoremap <leader>ft vatzf |
|  |  |
|  | " ,q Re-hardwrap Paragraph |
|  | nnoremap <leader>q gqip |
|  |  |
|  | " ,v Select just pasted text. |
|  | nnoremap <leader>v V`] |
|  |  |
|  | " ,ev Shortcut to edit .vimrc file on the fly on a vertical window. |
|  | nnoremap <leader>ev <C-w><C-v><C-l>:e $MYVIMRC<cr> |
|  |  |
|  | " jj For Qicker Escaping between normal and editing mode. |
|  | "inoremap jj <ESC> |
|  |  |
|  |  |
|  | " Working with split screen nicely |
|  | " Resize Split When the window is resized" |
|  | au VimResized \* :wincmd = |
|  |  |
|  |  |
|  | " Wildmenu completion " |
|  | set wildmenu |
|  | set wildmode=list:longest |
|  | set wildignore+=.hg,.git,.svn " Version Controls" |
|  | set wildignore+=\*.aux,\*.out,\*.toc "Latex Indermediate files" |
|  | set wildignore+=\*.jpg,\*.bmp,\*.gif,\*.png,\*.jpeg "Binary Imgs" |
|  | set wildignore+=\*.o,\*.obj,\*.exe,\*.dll,\*.manifest "Compiled Object files" |
|  | set wildignore+=\*.spl "Compiled speolling world list" |
|  | set wildignore+=\*.sw? "Vim swap files" |
|  | set wildignore+=\*.DS\_Store "OSX SHIT" |
|  | set wildignore+=\*.luac "Lua byte code" |
|  | set wildignore+=migrations "Django migrations" |
|  | set wildignore+=\*.pyc "Python Object codes" |
|  | set wildignore+=\*.orig "Merge resolution files" |
|  |  |
|  |  |
|  | " Make Sure that Vim returns to the same line when we reopen a file" |
|  | augroup line\_return |
|  | au! |
|  | au BufReadPost \* |
|  | \ if line("'\"") > 0 && line("'\"") <= line("$") | |
|  | \ execute 'normal! g`"zvzz' | |
|  | \ endif |
|  | augroup END |
|  |  |
|  | nnoremap g; g;zz |
|  |  |
|  | " =========== END Basic Vim Settings =========== |
|  |  |
|  |  |
|  | " =========== Gvim Settings ============= |
|  |  |
|  | " Removing scrollbars |
|  | if has("gui\_running") |
|  | set guitablabel=%-0.12t%M |
|  | set guioptions-=T |
|  | set guioptions-=r |
|  | set guioptions-=L |
|  | set guioptions+=a |
|  | set guioptions-=m |
|  | colo badwolf |
|  | set listchars=tab:▸\ ,eol:¬ " Invisibles using the Textmate style |
|  | else |
|  | set t\_Co=256 |
|  | colorschem badwolf |
|  | endif |
|  |  |
|  | " Special Settings for Consoles |
|  | "if !has("gui\_running") |
|  | " set t\_Co=256 |
|  | " colorschem badwolf |
|  | "endif |
|  | " Source the vimrc file after saving it |
|  | "autocmd bufwritepost .vimrc source ~/.vimrc |
|  |  |
|  | " ========== END Gvim Settings ========== |
|  |  |
|  |  |
|  | " ========== Plugin Settings ==========" |
|  |  |
|  | " Mapping to NERDTree |
|  | nnoremap <C-n> :NERDTreeToggle<cr> |
|  | let NERDTreeIgnore=['\.vim$', '\~$', '\.pyc$'] |
|  |  |
|  | " Mini Buffer some settigns." |
|  | let g:miniBufExplMapWindowNavVim = 1 |
|  | let g:miniBufExplMapWindowNavArrows = 1 |
|  | let g:miniBufExplMapCTabSwitchBufs = 1 |
|  | let g:miniBufExplModSelTarget = 1 |
|  |  |
|  | " Rope Plugin settings |
|  | imap <leader>j <ESC>:RopeGotoDefinition<cr> |
|  | nmap <leader>j <ESC>:RopeGotoDefinition<cr> |
|  |  |
|  | " Tagbar key bindings." |
|  | nmap <leader>l <ESC>:TagbarToggle<cr> |
|  | imap <leader>l <ESC>:TagbarToggle<cr>i |
|  |  |
|  | " Snipmate remap settings. It's binding overwritten standard |
|  | " bindings. |
|  | " Snipmate using bellow VIM's default keys, just unmap it. |
|  |  |
|  |  |
|  |  |
|  | " =========== END Plugin Settings ==========" |
|  | " |

<https://github.com/vim-airline/vim-airline>

# [vim-airline](https://github.com/vim-airline)/[vim-airline](https://github.com/vim-airline/vim-airline)

Plugin 'vim-airline/vim-airline' 

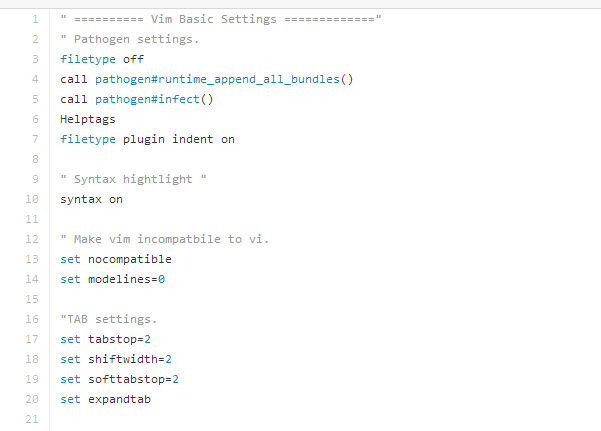
<https://github.com/sjl/badwolf>

# [sjl](https://github.com/sjl)/[badwolf](https://github.com/sjl/badwolf)

A color scheme for Vim, pieced together by [Steve Losh](http://stevelosh.com/).

[viruschidai](https://github.com/viruschidai)/[vim-as-nodejs-ide](https://github.com/viruschidai/vim-as-nodejs-ide)

<https://github.com/viruschidai/vim-as-nodejs-ide/blob/master/vim-files/vim/vimrc>



# <https://github.com/fholgado/minibufexpl.vim>

# [fholgado](https://github.com/fholgado)/[minibufexpl.vim](https://github.com/fholgado/minibufexpl.vim)

Plugin 'fholgado/minibufexpl.vim' 

|  |
| --- |
|  |
| " Mini Buffer some settigns." |
|  | let g:miniBufExplMapWindowNavVim = 1 |
|  | let g:miniBufExplMapWindowNavArrows = 1 |
|  | let g:miniBufExplMapCTabSwitchBufs = 1 |
|  | let g:miniBufExplModSelTarget = 1 |
|  |  |

<https://github.com/scrooloose/nerdtree>

# [scrooloose](https://github.com/scrooloose)/[nerdtree](https://github.com/scrooloose/nerdtree)

" Mapping to NERDTree

"| nnoremap <C-n> :NERDTreeToggle<cr>

" nnoremap <C-n> :NERDTreeToggle<cr>

" > because vim-multiple-cursors user (let g:multi\_cursor\_next\_key='<C-n>'), so use <C-t> for 'NERDTreeToggle'

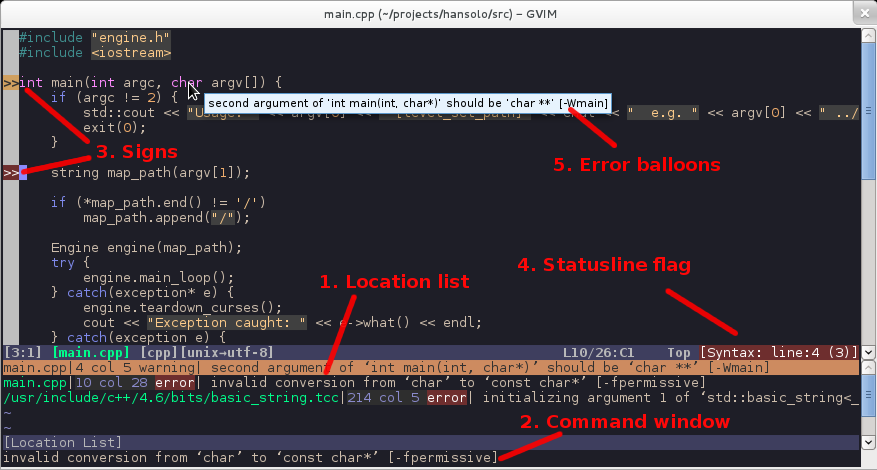
nnoremap <C-t> :NERDTreeToggle<cr>

let NERDTreeIgnore=['\.vim$', '\~$', '\.pyc$']

<https://github.com/scrooloose/syntastic>

# [scrooloose](https://github.com/scrooloose)/[syntastic](https://github.com/scrooloose/syntastic)

Syntastic is a syntax checking plugin for [Vim](http://www.vim.org/) that runs files through external syntax checkers and displays any resulting errors to the user.



" syntastic

set statusline+=%#warningmsg#

set statusline+=%{SyntasticStatuslineFlag()}

set statusline+=%\*

"| let g:syntastic\_always\_populate\_loc\_list = 1

let g:syntastic\_always\_populate\_loc\_list = 0

"| let g:syntastic\_auto\_loc\_list = 1

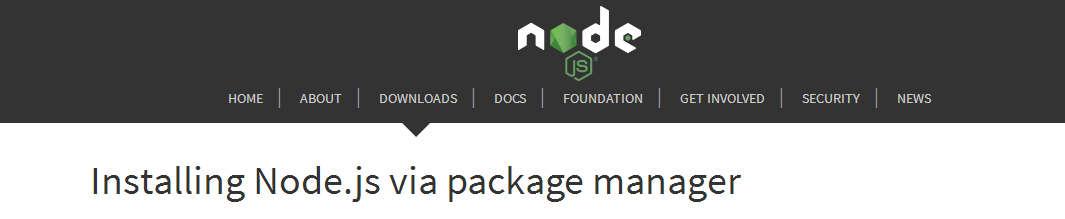
let g:syntastic\_auto\_loc\_list = 0

let g:syntastic\_check\_on\_open = 1

let g:syntastic\_check\_on\_wq = 0

<https://nodejs.org/en/download/package-manager/#debian-and-ubuntu-based-linux-distributions>

# Installing Node.js via package manager



sudo apt-get install curl

curl -sL https://deb.nodesource.com/setup\_4.x | sudo -E bash -

sudo apt-get install -y nodejs

jshint2.vim插件需要先（依赖）安装node和jshint

<https://github.com/Shutnik/jshint2.vim>

# [Shutnik](https://github.com/Shutnik)/[jshint2.vim](https://github.com/Shutnik/jshint2.vim)

# jshint2.vim

**使用npm安装一些包失败了的看过来（npm国内镜像介绍）**

 发布于 4 年前  作者 [wppept](https://cnodejs.org/user/wppept)  129790 次浏览  最后一次编辑是 2 年前

这个也是网上搜的，亲自试过，非常好用！

镜像使用方法（三种办法任意一种都能解决问题，建议使用第三种，将配置写死，下次用的时候配置还在）:

1.通过config命令

npm config set registry https://registry.npm.taobao.org

npm info underscore （如果上面配置正确这个命令会有字符串response）

2.命令行指定

npm --registry https://registry.npm.taobao.org info underscore

3.编辑 ~/.npmrc 加入下面内容

registry = https://registry.npm.taobao.org

<http://jshint.com/install/>

sudo npm install -g jshint

### Plugins for text editors and IDEs

#### VIM

* [jshint.vim](https://github.com/walm/jshint.vim), VIM plugin and command line tool for running JSHint.
* [jshint2.vim](https://github.com/Shutnik/jshint2.vim), modern VIM plugin with extra features for running JSHint.
* [Syntastic](https://github.com/scrooloose/syntastic), supports JSHint both older/newer than 1.1.0.

# [sheerun](https://github.com/sheerun)/[vim-polyglot](https://github.com/sheerun/vim-polyglot)

<https://github.com/sheerun/vim-polyglot>

A solid language pack for Vim.

**Troubleshooting**

Please make sure you have syntax on in your .vimrc, otherwise syntax files are not loaded at all.

Individual language packs can be disabled by setting g:polyglot\_disabled as follows:

" ~/.vimrc

let g:polyglot\_disabled = ['css']

# [majutsushi](https://github.com/majutsushi)/[tagbar](https://github.com/majutsushi/tagbar)

<https://majutsushi.github.io/tagbar/>

tagbar报错的问题的解决办法

<http://stackoverflow.com/questions/7454796/taglist-exuberant-ctags-not-found-in-path>

# [Taglist: Exuberant ctags not found in PATH](http://stackoverflow.com/questions/7454796/taglist-exuberant-ctags-not-found-in-path)

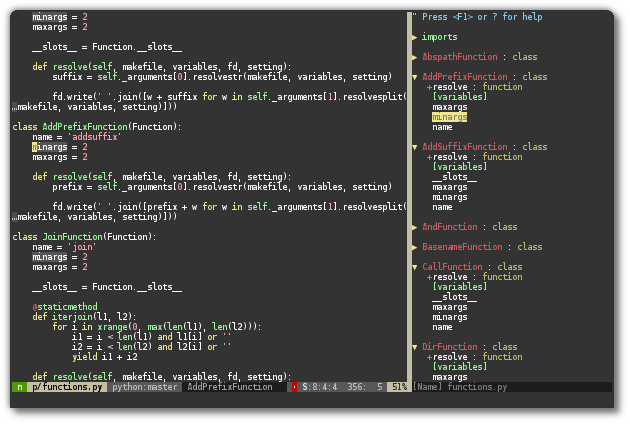
For Ubuntu and derivatives:

sudo apt-get install exuberant-ctags

With yum:

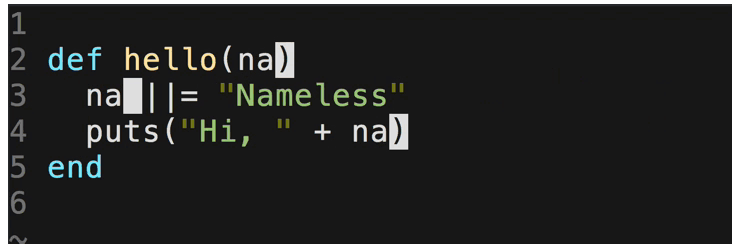
sudo yum install ctags-etags

|  |
| --- |
|  |
| " Tagbar key bindings." |
|  | nmap <leader>l <ESC>:TagbarToggle<cr> |
|  | imap <leader>l <ESC>:TagbarToggle<cr>i |
|  |  |



# [terryma](https://github.com/terryma)/[vim-multiple-cursors](https://github.com/terryma/vim-multiple-cursors)

<https://github.com/terryma/vim-multiple-cursors>



You can then map the 'next', 'previous', 'skip', and 'exit' keys like the following:

" Default mapping

let g:multi\_cursor\_next\_key='<C-n>'

let g:multi\_cursor\_prev\_key='<C-p>'

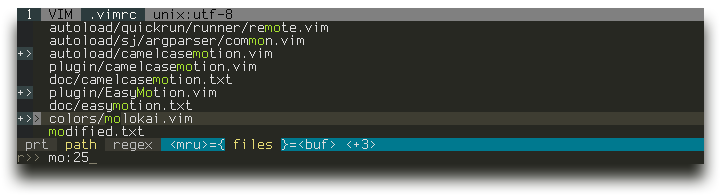
let g:multi\_cursor\_skip\_key='<C-x>'

let g:multi\_cursor\_quit\_key='<Esc>'

# [ctrlpvim](https://github.com/ctrlpvim)/[ctrlp.vim](https://github.com/ctrlpvim/ctrlp.vim)

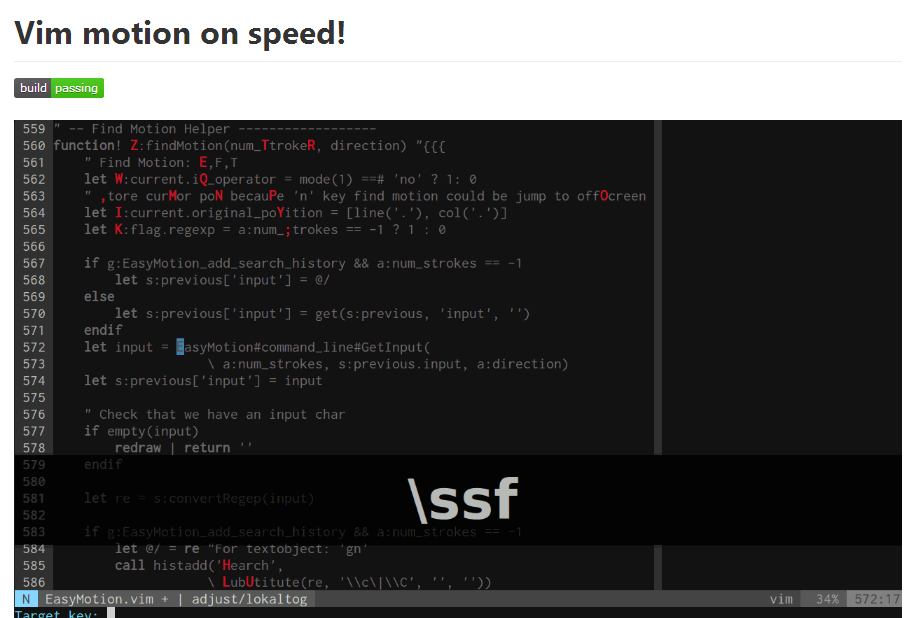
<https://github.com/ctrlpvim/ctrlp.vim>

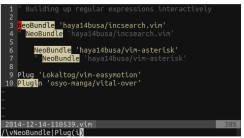
Full path fuzzy **file**, **buffer**, **mru**, **tag**, **...** finder for Vim.

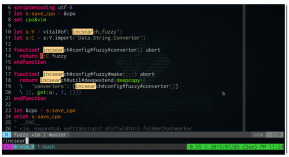


# [easymotion](https://github.com/easymotion)/[vim-easymotion](https://github.com/easymotion/vim-easymotion)

<https://github.com/easymotion/vim-easymotion>



<https://github.com/haya14busa/incsearch.vim>

<https://github.com/haya14busa/incsearch-easymotion.vim>

* [haya14busa/incsearch.vim](https://github.com/haya14busa/incsearch.vim)
* [haya14busa/incsearch-easymotion.vim](https://github.com/haya14busa/incsearch-easymotion.vim)

#### Integration with incsearch.vim

* [haya14busa/incsearch.vim](https://github.com/haya14busa/incsearch.vim)
* [haya14busa/incsearch-easymotion.vim](https://github.com/haya14busa/incsearch-easymotion.vim)

" You can use other keymappings like <C-l> instead of <CR> if you want to

" use these mappings as default search and somtimes want to move cursor with

" EasyMotion.

function! s:incsearch\_config(...) abort

return incsearch#util#deepextend(deepcopy({

\ 'modules': [incsearch#config#easymotion#module({'overwin': 1})],

\ 'keymap': {

\ "\<CR>": '<Over>(easymotion)'

\ },

\ 'is\_expr': 0

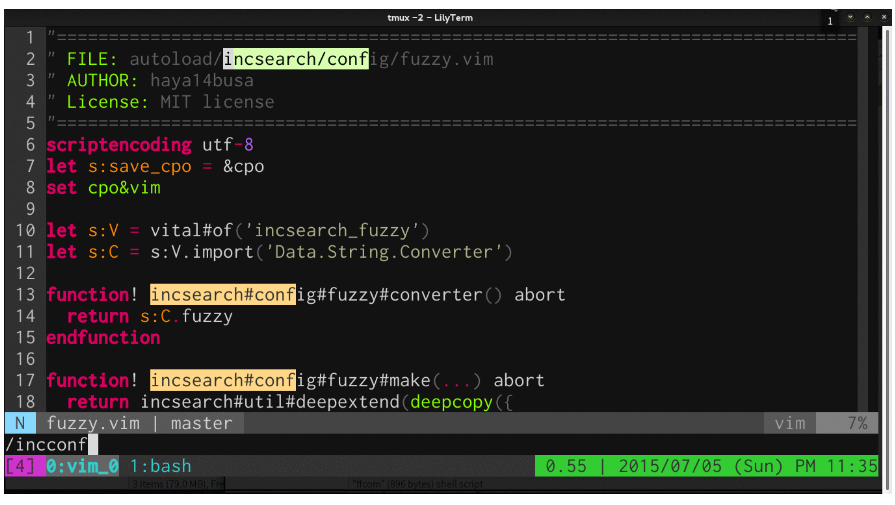
\ }), get(a:, 1, {}))

endfunction

noremap <silent><expr> / incsearch#go(<SID>incsearch\_config())

noremap <silent><expr> ? incsearch#go(<SID>incsearch\_config({'command': '?'}))

noremap <silent><expr> g/ incsearch#go(<SID>incsearch\_config({'is\_stay': 1}))



* [haya14busa/incsearch.vim](https://github.com/haya14busa/incsearch.vim)
* [haya14busa/incsearch-fuzzy.vim](https://github.com/haya14busa/incsearch-fuzzy.vim)

function! s:config\_easyfuzzymotion(...) abort

return extend(copy({

\ 'converters': [incsearch#config#fuzzyword#converter()],

\ 'modules': [incsearch#config#easymotion#module({'overwin': 1})],

\ 'keymap': {"\<CR>": '<Over>(easymotion)'},

\ 'is\_expr': 0,

\ 'is\_stay': 1

\ }), get(a:, 1, {}))

endfunction

noremap <silent><expr> <Space>/ incsearch#go(<SID>config\_easyfuzzymotion())

" Gif config

map / <Plug>(easymotion-sn)

omap / <Plug>(easymotion-tn)

" These `n` & `N` mappings are options. You do not have to map `n` & `N` to EasyMotion.

" Without these mappings, `n` & `N` works fine. (These mappings just provide

" different highlight method and have some other features )

" map n <Plug>(easymotion-next)

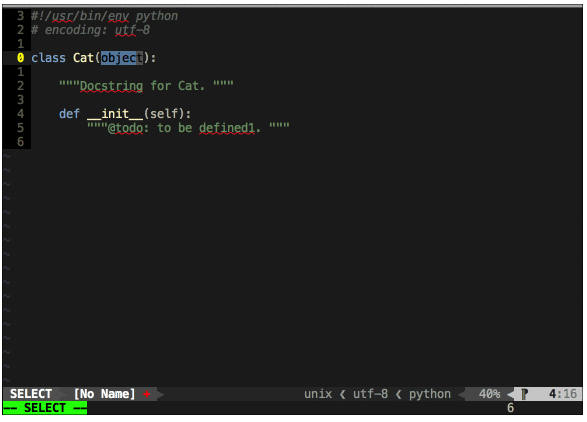
" map N <Plug>(easymotion-prev)

# UltiSnips

<https://github.com/SirVer/ultisnips>

UltiSnips is the ultimate solution for snippets in Vim. It has tons of features and is very fast.

# [honza](https://github.com/honza)/[vim-snippets](https://github.com/honza/vim-snippets)



# [honza](https://github.com/honza)/[vim-snippets](https://github.com/honza/vim-snippets)

<https://github.com/honza/vim-snippets>

" Track the engine.

Plugin 'SirVer/ultisnips'

" Snippets are separated from the engine. Add this if you want them:

Plugin 'honza/vim-snippets'

" Ultisnips

" Trigger configuration. Do not use <tab> if you use https://github.com/Valloric/YouCompleteMe.

" http://stackoverflow.com/questions/14896327/ultisnips-and-youcompleteme

" SnipMate/UltiSnips integration #36

" https://github.com/Valloric/YouCompleteMe/issues/36

let g:UltiSnipsExpandTrigger="<c-j>"

"let g:UltiSnipsExpandTrigger="<tab>"

"let g:UltiSnipsJumpForwardTrigger="<c-b>"

"let g:UltiSnipsJumpBackwardTrigger="<c-z>"

" If you want :UltiSnipsEdit to split your window.

let g:UltiSnipsEditSplit="vertical"

# [Valloric](https://github.com/Valloric)/[YouCompleteMe](https://github.com/Valloric/YouCompleteMe)

<https://github.com/Valloric/YouCompleteMe>

<https://valloric.github.io/YouCompleteMe/#ubuntu-linux-x64>

### Ubuntu Linux x64

Make sure you have Vim 7.3.598 with python2 or python3 support. Ubuntu 14.04 and later have a Vim that's recent enough. You can see the version of Vim installed by running vim --version. If the version is too old, you may need to [compile Vim from source](https://github.com/Valloric/YouCompleteMe/wiki/Building-Vim-from-source) (don't worry, it's easy).

Install development tools and CMake: sudo apt-get install build-essential cmake

Make sure you have Python headers installed: sudo apt-get install python-dev python3-dev.

Compiling YCM **with** semantic support for C-family languages:

# [ternjs](https://github.com/ternjs)/[tern\_for\_vim](https://github.com/ternjs/tern_for_vim)

## Installation

#### Manual

If you use [Pathogen](https://github.com/tpope/vim-pathogen) or something similar, you can clone this repository to your ~/.vim/bundle (or equivalent) directory. Make sure you have [node.js](http://nodejs.org/) and [npm](https://npmjs.org/) installed (Tern is a JavaScript program), and install the tern server by running npm install in the bundle/tern\_for\_vim directory.

**Caution**: Because the node process is not run using your standard shell, the NVM version of node.js won't work. You need a global node executable.

## Configuration

The command used to start the Tern server can be overridden by setting tern#command to an array of strings (the binary and its arguments, if any). You might need this if your node is installed somewhere that's not in the default path, or if you want to install Tern in some custom location.

The variable tern#is\_show\_argument\_hints\_enabled can be set to something truthy to make the plugin display the arguments to the current function at the bottom of the screen. This feature is currently not very mature, and likely to make your editing laggy.

Tern uses .tern-project files to configure loading libraries and plugins for a project. See the [Tern docs](http://ternjs.net/doc/manual.html#configuration) for details.

# Tern＋YouCompleteMe实现vim中JS自动补全

<http://www.jianshu.com/p/4a8b0e3503fa>

# 配置 Tern

1，在Tern目录下执行npm install

cd ~/.vim/bundle/tern\_for\_vim

npm install

2， 在你的项目根目录创建 .tern\_project 文件，并配置

touch .tern\_project

在其中加入如下内容

{

  "libs": [

    "browser",

    "underscore",

    "jquery"

  ],

  "plugins": {

  }

}

在plugin中可以加入node或者angular等关键字，加载该库的补全功能。如下：

"plugins": {

  "node": {}

}

接下来便可以进行使用了，打开项目中的文件，感受自动补全的快感～

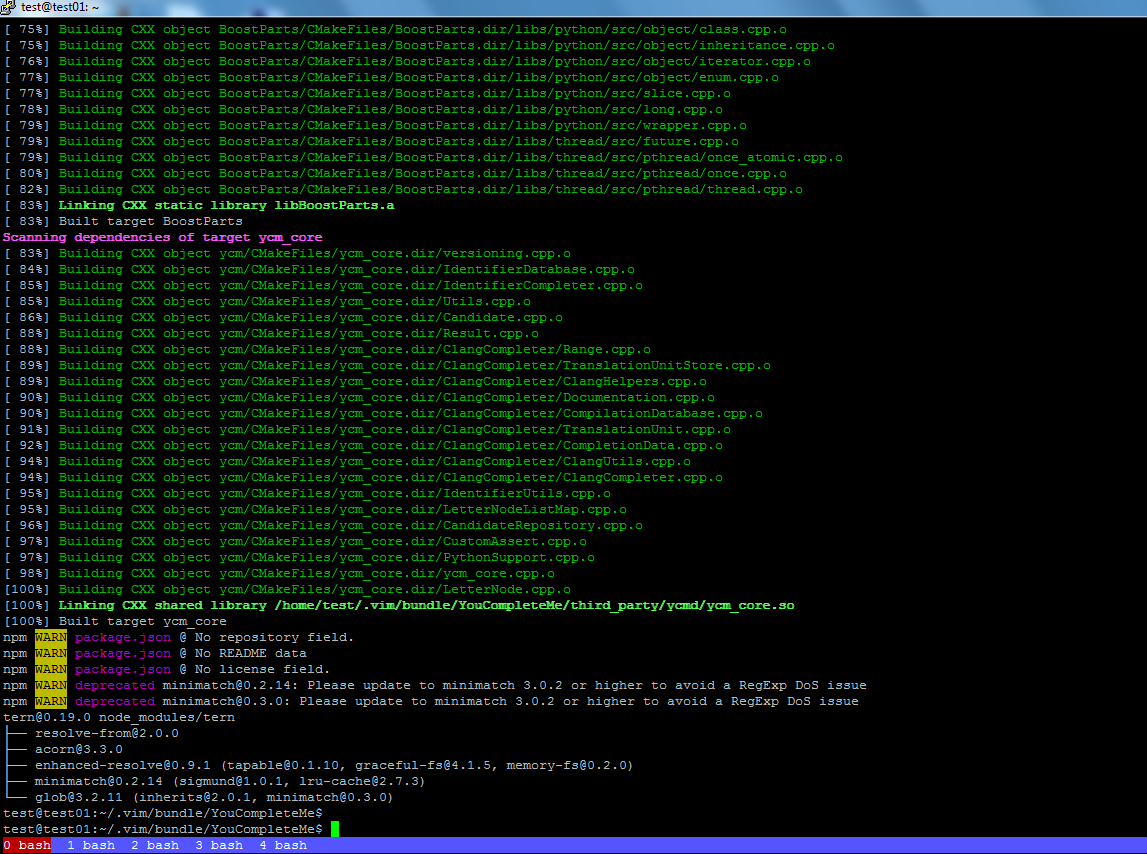
文／WillDad（简书作者）  
原文链接：http://www.jianshu.com/p/4a8b0e3503fa  
著作权归作者所有，转载请联系作者获得授权，并标注“简书作者”。

* JavaScript support: install [Node.js and npm](https://docs.npmjs.com/getting-started/installing-node) and add --tern-completer when calling ./install.py.

cd ~/.vim/bundle/YouCompleteMe

git submodule update --init --recursive

./install.py --clang-completer --tern-completer



<https://www.npmjs.com/package/tern>

# [tern](https://www.npmjs.com/package/tern)

A JavaScript code analyzer for deep, cross-editor language support

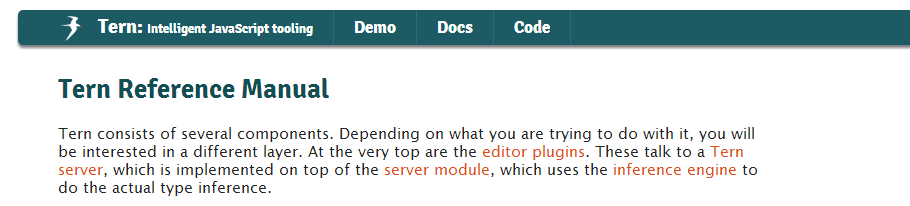
This is [**Tern**](http://ternjs.net/). Tern is a stand-alone, editor-independent JavaScript analyzer that can be used to improve the JavaScript integration of existing editors.

Thanks to a group of generous [**crowd funders**](http://www.indiegogo.com/projects/tern-intelligent-javascript-editing), Tern is open-source software, under an MIT license.

There are currently plugins available for [**Emacs**](http://ternjs.net/doc/manual.html#emacs) (and Emacs [**company-mode**](https://github.com/proofit404/company-tern)), [**Vim**](https://github.com/ternjs/tern_for_vim), [**Sublime Text**](https://github.com/ternjs/tern_for_sublime), [**Eclipse (and general Java API)**](https://github.com/angelozerr/tern.java), [**Light Table**](https://github.com/mortalapeman/LT-TernJS), [**Atom**](https://atom.io/packages/atom-ternjs), [**TextMate**](https://github.com/fab1an/JavaScript-Tern-Completion.tmbundle) and [**gedit**](https://github.com/Swatinem/tern_for_gedit), and built-in support in [**Brackets**](http://brackets.io/), [**Edge Code**](http://html.adobe.com/edge/code), [**CodeLite**](http://codelite.org/), and [**vy**](https://github.com/iogf/vy).

For further documentation, see the [**project page**](http://ternjs.net/) and the [**manual**](http://ternjs.net/doc/manual.html). To report issues, use the [**issue tracker**](https://github.com/ternjs/tern/issues). For questions and documentation, see the [**discussion forum**](https://discuss.ternjs.net/).

<http://ternjs.net/doc/manual.html>



When started, the server will look for a .tern-project file in the current directory or one of the directories above that, and use it for its [configuration](http://ternjs.net/doc/manual.html#configuration). If no project file is found, it’ll fall back to a default configuration. You can change this default configuration by putting a .tern-config file, with the same format as .tern-project, in your home directory.

### Project configuration

A .tern-project file is a [JSON](http://json.org/) file in a format like this:

{

"libs": [

"browser",

"jquery"

],

"loadEagerly": [

"importantfile.js"

],

"plugins": {

"requirejs": {

"baseURL": "./",

"paths": {}

}

}

}

The libs property refers to the [JSON type descriptions](http://ternjs.net/doc/manual.html#typedef) that should be loaded into the environment for this project. See the defs/ directory for examples. The strings given here will be suffixed with .json, and searched for first in the project’s own dir, and then in the defs/directory of the Tern distribution.

By default, local files are loaded into the Tern server when queries are run on them in the editor. loadEagerly allows you to force some files to always be loaded, it may be an array of filenames or glob patterns (i.e. foo/bar/\*.js). The dontLoad option can be used to prevent Tern from loading certain files. It also takes an array of file names or glob patterns.

The plugins field may hold object used to load and configure Tern plugins. The names of the properties refer to files that implement plugins, either in the project dir or under plugin/ in the Tern directory. Their values are configuration objects that will be passed to the plugins. You can leave them at {} when you don’t need to pass any options.

You can specify an ecmaVersion field to configure the version of ECMAScript that Tern parses. The default is 6, and leaving it at that should be safe even for ECMAScript 5 code, but you can set it to 5 as well.

To configure the amount of work Tern is prepared to do to load a single dependency, thedependencyBudget option can be added to a project file. It indicates the maximum size of the files loaded in response to a single dependency (through plugins that load dependencies, such as the [node](http://ternjs.net/doc/manual.html#plugin_node) and [RequireJS](http://ternjs.net/doc/manual.html#plugin_requirejs) plugins), counted in expressions. The default value is 20 000. Files loaded as dependencies of dependencies count towards the budget of the original dependency.

# [How to navigate a JavaScript project?](http://emacs.stackexchange.com/questions/3093/how-to-navigate-a-javascript-project)

<http://emacs.stackexchange.com/questions/3093/how-to-navigate-a-javascript-project>

可以将下面的内容放到家目录中的**.tern-config** 文件中作为默认配置（但是我将"src/\*\*.js"注释掉了）

The final step is to set up a .tern-project file *in the root directory of your project*. For Node.js projects, I use this as a starting point:

{

"loadEagerly": [

"src/\*\*.js"

],

"plugins": {

"node": {

}

}

}

# Tern＋YouCompleteMe实现vim中JS自动补全

<http://www.jianshu.com/p/4a8b0e3503fa>

# windows下安装tern for vim

<http://www.tuicool.com/articles/3UF3AbI>

打开qianye.js，输入jQuery.，按ctrl+x和ctrl+o，是不是有提示了。

A code-completion engine for Vim <http://valloric.github.io/YouCompleteMe/>

按ctrl-x和ctrl-o进入提示模式。

**YouCompleteMe常用快捷键：**

**按ctrl-x和ctrl-o进入提示模式。**

**ctrl + n**

**ctrl + p**

**ctrl + j "在 Ultisnips中配置的let g:UltiSnipsExpandTrigger="<c-j>"，对提示中的snippet使用**

**ctrl + y**

**"YouCompleteMe的配置======.vimrc=====开始=================================**

let g:ycm\_key\_invoke\_completion = '<M-Space>'

let g:ycm\_filetype\_blacklist = {

\ 'tagbar' : 1,

\ 'qf' : 1,

\ 'notes' : 1,

\ 'markdown' : 1,

\ 'unite' : 1,

\ 'text' : 1,

\ 'vimwiki' : 1,

\ 'pandoc' : 1,

\ 'infolog' : 1,

\ 'mail' : 1,

\ 'nerdtree' : 1,

\}

" I get a weird window at the top of my file when I use the semantic engine

"If you don't want this window to ever show up, add set completeopt-=preview to your vimrc. Also make sure that the g:ycm\_add\_preview\_to\_completeopt option is set to 0.

" https://github.com/Valloric/YouCompleteMe ----------------------

"let g:ycm\_autoclose\_preview\_window\_after\_insertion = 0

"let g:ycm\_autoclose\_preview\_window\_after\_completion = 0

let g:ycm\_autoclose\_preview\_window\_after\_completion = 1

let g:ycm\_autoclose\_preview\_window\_after\_insertion = 1

set completeopt-=preview

let g:ycm\_add\_preview\_to\_completeopt = 0

" http://stackoverflow.com/questions/14896327/ultisnips-and-youcompleteme

" UltiSnips and YouCompleteMe

" if you use Vundle, load plugins:

" Plugin 'ervandew/supertab'

" Plugin 'Valloric/YouCompleteMe'

" Plugin 'SirVer/ultisnips'

" make YCM compatible with UltiSnips (using supertab)

let g:ycm\_key\_list\_select\_completion = ['<C-n>', '<Down>']

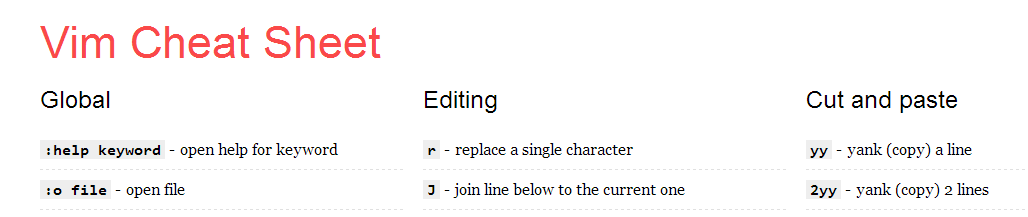
let g:ycm\_key\_list\_previous\_completion = ['<C-p>', '<Up>']

let g:SuperTabDefaultCompletionType = '<C-n>'

**"YouCompleteMe的配置======.vimrc=====结束=================================**

Vim Cheat Sheet

<http://vim.rtorr.com/>



<http://coolshell.cn/articles/5426.html>

**简明 Vim 练级攻略**



<http://askubuntu.com/questions/863/change-primary-monitor>

# [Change Primary monitor](http://askubuntu.com/questions/863/change-primary-monitor)

Until Ubuntu Brainstorm Idea #17526 becomes a reality, it seems there is no way for non-NVIDIA users to change the primary display (not just move the panels) without resorting to the command line.

Edit 2014-05-30: That answer is a little outdated now. Ubuntu Brainstorm no longer exists. And other answers to this question have better solutions, both for Ubuntu, and other distros.

In MATE, "Preferences > Monitors" has a "Make Default" button. That's what you're looking for.

In GNOME 3, "Settings > Display" doesn't have a "Make Default" button. Instead it has a toolbar-looking bar at the top of one of the displays. Drag that to the display that you want to be primary.

I believe Unity is similar to GNOME 3, but the toolbar is on the side -- I don't know for sure though; I'm not running Unity myself.

In KDE, under "System Settings > Display and Monitor" there is a star on each display that you can click to set which is primary.

" ========================================

" Better colors for syntax highlighting

" If you have a black background, use the following command to get a better color map for syntax highlighting:

set background=dark

" This function reverses background setting so you can quickly toggle between light and dark .

function! ReverseBackground()

let Mysyn=&syntax

if &bg=="light"

se bg=dark

else

se bg=light

endif

syn on

exe "set syntax=" . Mysyn

echo "now syntax is "&syntax

endfunction

command! Invbg call ReverseBackground()

noremap <F11> :Invbg<CR>

" ========================================

解决screen中不能拖拽鼠标来resize window大小的问题

" > -----------------http://superuser.com/questions/549930/cant-resize-vim-splits-inside-tmux-----------------

" > -----------------enable split window can resize by dragging mouse in some programm context(eg: the 'screen' program window)-----------------

" Enable Mouse

set mouse=a

" >>>>>>>> note: the following setting can work on ubuntu but CentOS6.x. the following setting prevents the scrolling up and down in the right way on CentOS

set mouse+=a

if &term =~ '^screen'

" tmux knows the extended mouse mode

set ttymouse=xterm2

endif

" <<<<<<<<

" >>>>>>> enable the following setting on CentOS6.x and disable the previous setting that can work on Ubuntu.

" if &term =~ '^screen'

" " tmux knows the extended mouse mode

" set ttymouse=xterm2

" endif

" <<<<<<<

<https://github.com/godlygeek/tabular>

# [godlygeek](https://github.com/godlygeek)/[tabular](https://github.com/godlygeek/tabular)

<http://vimcasts.org/episodes/aligning-text-with-tabular-vim/>

Aligning text with Tabular.vim

**:Tab /=**

**:Tab/:**

**:Tab /:\zs**

Examples:

|start|eat|left|

|12|5|7|

|20|5|15|

With the cursor positioned anywhere in the table, running **:Tab/|** produces:

Examples:

| start | eat | left |

| 12 | 5 | 7 |

| 20 | 5 | 15 |

vim格式化工具：Tabularize 使用指南

<http://my.oschina.net/huangsz/blog/176120>

**安装微软的Courier New字体：**

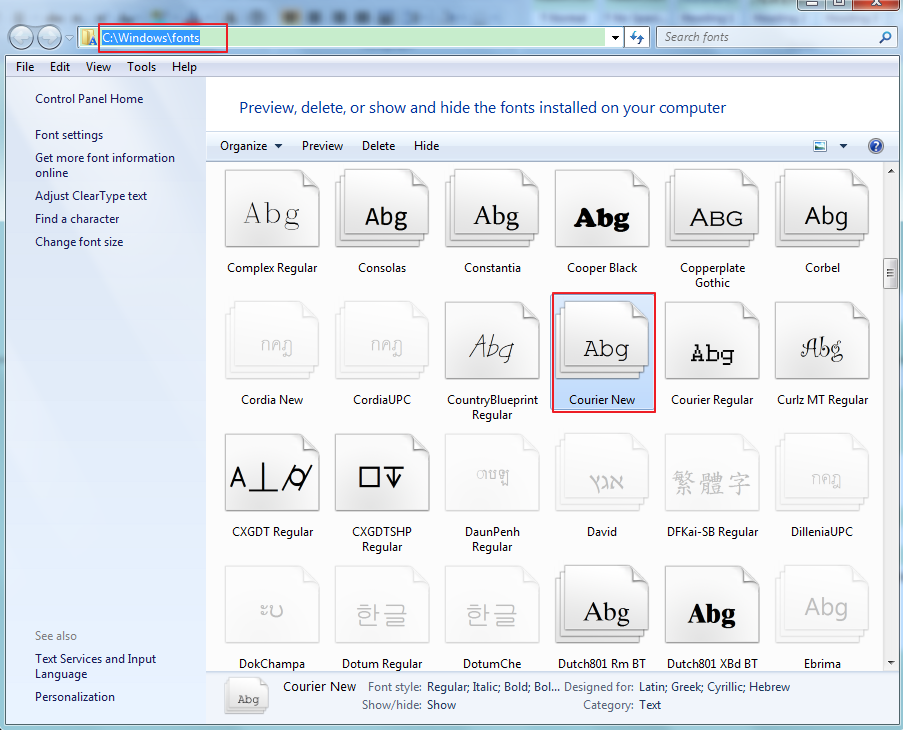
在win7系统下执行：

<https://support.microsoft.com/en-us/kb/314960>

首先在windows 7系统下找到Courier New字体文件，

1. Click **Start**, and then click **Run**.
2. Type the following command, and then click **OK**:

%windir%\fonts



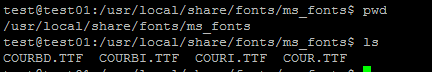
# How to Manually Install Fonts in Ubuntu 16.04

<http://ubuntuhandbook.org/index.php/2016/05/manually-install-fonts-ubuntu-16-04/>

**2.** In Ubuntu Linux, font files are installed to **/usr/lib/share/fonts** or **/usr/share/fonts**. The former directory is recommended in this case for manual installation.

First create a sub folder in the font directory, ms\_fonts, as example:

sudo mkdir /usr/local/share/fonts/ms\_fonts #然后将Courier New字体文件文件copy到该目录下，结果如下图：



Type in your password when it asks (no visual feedback) and hit Enter.

**3.** Move all font files from your Downloads directory to the new created folder:

sudo mv ~/Downloads/\*.ttc ~/Downloads/\*.ttf /usr/local/share/fonts/ms\_fonts/

**4.** Set permissions to these files by running the commands below one by one:

sudo chown root:staff /usr/local/share/fonts/ms\_fonts -R

sudo chmod 644 /usr/local/share/fonts/ms\_fonts/\* -R

sudo chmod 755 /usr/local/share/fonts/ms\_fonts

**5.** Finally builds font information caches for apps using fontconfig for their font handling:

sudo fc-cache -fv

<http://theholmesoffice.com/how-to-share-folders-between-windows-and-ubuntu-using-vmware-player/>

[SUBSCRIBE](http://theholmesoffice.com/rss/)

# How to share folders between Windows and Ubuntu using VMware Player

<https://pubs.vmware.com/workstation-9/index.jsp?topic=%2Fcom.vmware.ws.using.doc%2FGUID-D6D9A5FD-7F5F-4C95-AFAB-EDE9335F5562.html>

**Enable a Shared Folder for a Virtual Machine**

**Prerequisites**

|  |  |
| --- | --- |
| ■ | Verify that the virtual machines use a guest operating system that supports shared folders. See [Guest Operating Systems that Support Shared Folders](https://pubs.vmware.com/workstation-9/topic/com.vmware.ws.using.doc/GUID-7DAD8EC0-0403-4152-86ED-99700F6C02B8.html#GUID-7DAD8EC0-0403-4152-86ED-99700F6C02B8). |
| ■ | Verify that the latest version of VMware Tools is installed in the guest operating system. |
| ■ | Verify that permission settings on the host system allow access to files in the shared folders. For example, if you are running Workstation as a user named User, the virtual machine can read and write files in the shared folder only if User has permission to read and write them. |

<http://www.omgubuntu.co.uk/2016/04/10-things-to-do-after-installing-ubuntu-16-04-lts>

## 16 Things To Do After Installing Ubuntu 16.04 LTS

<http://www.tecmint.com/useful-basic-commands-of-apt-get-and-apt-cache-for-package-management/>

# 25 Useful Basic Commands of APT-GET and APT-CACHE for Package Management

<http://iask.sina.com.cn/b/9093431.html>

### 电脑显示器离人眼睛多远最好

为了保护视力显示器应放距离眼睛多远？

1.简单判断办法：要检测显示器是否摆放恰当，只需做一个简单的测试。首先，在显示器上确定一点，然后闭上眼睛，头部慢慢地做几次上下运动。当头部运动到感觉舒服的位置时，睁开眼睛。如果显示器摆放合适的话，眼睛应刚好定格在之前记下的那一点上。

2.经验数据：显示器和人之间的距离，应该根据其对角线长度来计算，大约是对角线长度的2倍！如果显示器对角线长35厘米，那眼睛和显示器之间的距离应不少于60厘米；如果对角线长38厘米以上，那眼睛和它的距离应不少于70厘米。

■距离眼睛水平位置多高为宜？

显示器应该摆在人身体的正前方、视线稍下方。如果显示器位置较高，人眼处于仰视时，睁开最大，眼球表面充分暴露在空气中，时间久了容易眼干。此外，他也不建议将显示器朝前掰，这样图像会不清晰，从而影响视力。一般情况下，那些坐得特别高的人，只需将显示器往后掰一些，使屏幕和人的面部处于平行位置，就可以了。”