```
"unbind space and remap it to mapleader
let mapleader = " "


"_____________________________________________________________________
_____
"VIM PLUG
"specify a directory for pluggins
call plug#begin('~/.vim/plugged')

"On Demand Loading
Plug 'scrooloose/nerdtree', {'on': 'NERDTreeToggle'}

"Auto Closing
Plug 'alvan/vim-closetag'
Plug 'jiangmiao/auto-pairs'

"Comment Lines
Plug 'preservim/nerdcommenter'

"Status bar 'airline'
Plug 'vim-airline/vim-airline'
Plug 'vim-airline/vim-airline-themes'

"Buffer Bar
 Plug 'ap/vim-buftabline'

"Powerline/ Powerline fonts
Plug 'powerline/powerline'
Plug 'powerline/fonts'

"autoprefixer
Plug 'vim-scripts/vim-autoprefixer'

"COLORSCHEMES
Plug 'flazz/vim-colorschemes'

"COLOR HIGHLIGHTER HEX AND RGB COLOR DESIGNATIONS
" Plug 'ap/vim-css-color'
Plug 'shmargum/vim-sass-colors'

"Better Javascript syntax highlighting
" Plug 'pangloss/vim-javascript'

"syntax highlighting and support for all languages
Plug 'sheerun/vim-polyglot'

"autocomplete
Plug 'ajh17/VimCompletesMe'
```

```
"Vim Color Picker
Plug 'iandoe/vim-osx-colorpicker'
let g:colorpicker_app = 'iTerm.app'

"Vim CSS Color Highlighting
" Plug 'skammer/vim-css-color'
" let g:cssColorVimDoNotMessMyUpdatetime = 1


"End VIMPLUG
call plug#end()



"airline symbols
let g:airline_powerline_fonts = 1
set encoding=utf-8

if !exists('g:airline_symbols')
    let g:airline_symbols = {}
  endif


  " unicode symbols
  let g:airline_left_sep = '»'
  let g:airline_left_sep = '▶'
  let g:airline_right_sep = '«'
  let g:airline_right_sep = '◀'
  let g:airline_symbols.crypt = '🔒'
  let g:airline_symbols.linenr = '☰'
  let g:airline_symbols.linenr = '␊'
  let g:airline_symbols.linenr = '␤'
  let g:airline_symbols.linenr = '¶'
  let g:airline_symbols.maxlinenr = ''
  let g:airline_symbols.maxlinenr = '㏑'
  let g:airline_symbols.branch = '⎇'
  let g:airline_symbols.paste = 'ρ'
  let g:airline_symbols.paste = 'Þ'
  let g:airline_symbols.paste = '∥'
  let g:airline_symbols.spell = 'Ꞩ'
  let g:airline_symbols.notexists = 'Ɇ'
  let g:airline_symbols.whitespace = 'Ξ'

  " powerline symbols
  let g:airline_left_sep = '?'
  let g:airline_left_alt_sep = '?'
  let g:airline_right_sep = '?'
  let g:airline_right_alt_sep = '?'
```

```
  let g:airline_symbols.branch = '?'
  let g:airline_symbols.readonly = '?'
  let g:airline_symbols.linenr = '☰'
  let g:airline_symbols.maxlinenr = '?'
  let g:airline_symbols.dirty='⚡'

  " old vim-powerline symbols
  let g:airline_left_sep = '▶'
  let g:airline_left_alt_sep = '❯'
  let g:airline_right_sep = '◀'
  let g:airline_right_alt_sep = '❮'
  let g:airline_symbols.branch = '⭠'
  let g:airline_symbols.readonly = '⭤'
  let g:airline_symbols.linenr = '␊'

"_______________________________________________________________________
______
"KEY REMAPS
"remaps esc to 'kj' and stops the cursor from going back one space
when you
"exit insert mode.
inoremap <silent>  kj <Esc>`^

" Go to tab by number
noremap <leader>1 1gt
noremap <leader>2 2gt
noremap <leader>3 3gt
noremap <leader>4 4gt
noremap <leader>5 5gt
noremap <leader>6 6gt
noremap <leader>7 7gt
noremap <leader>8 8gt
noremap <leader>9 9gt
noremap <leader>0 :tablast<cr>

" Go to last active tab
au TabLeave * let g:lasttab = tabpagenr()
nnoremap <silent> <c-l> :exe "tabn ".g:lasttab<cr>
vnoremap <silent> <c-l> :exe "tabn ".g:lasttab<cr>

"Copy and Paste from system clipboard
noremap <Leader>y "*y
noremap <Leader>p "*p
noremap <Leader>Y "+y
noremap <Leader>P "+p

"Set new splits down and to the right
set splitbelow
set splitright
```

```
"map F5 to reset highlighting
" map <F5> :syntax sync fromstart<cr>
"map f12 to reset highlighting
noremap <F12> <Esc>:syntax sync fromstart<CR>
inoremap <F12> <C-o>:syntax sync fromstart<CR>

"set redraw time longer to give time for syntax highlighting
" set redrawtime=10000

"KEY REMAPS END
"------------------------------------------------------------------------
------

"------------------------------------------------------------------------
------
"STRANGE FIXES

"Allow plugins?
set nocompatible
filetype plugin on
runtime macros/machit.vim

"Turn on Backspace (not activated by default.)
set bs=2

"Fix Visual Mode Highlighting.
:highlight Visual cterm=reverse ctermbg=NONE

"SET TAB TO 2 SPACES
filetype plugin indent on
"show existing tab with 2 spaces width
set tabstop=2
"when indenting with '>' use 2 spaces width
set shiftwidth=2
"on pressing tab, insert 2 spaces
set expandtab
set softtabstop=2
"Stop @@@@@@@ from displaying after long last line
set display+=lastline

" FIXES FOR VIM COLORS
"set Vim-specific sequences for RGB colors
" set termguicolors

set t_Co=256

" set background=dark
" if (has("termguicolors"))
set termguicolors
" in order to reactivate the two items below (delete the '\' character
```

```
after
" the equals sign')
let &t_8f = "\<Esc>[38;2;%lu;%lu;%lum"
let &t_8b = "\<Esc>[48;2;%lu;%lu;%lum"

"Fix for mouse not working to resize windows
set ttymouse=xterm2
" set ttymouse=sgr

" endif

"FTP stuff
"Activate ftp passive mode in vim
let g:netrw_ftp_cmd= "ftp -p"

"STRANGE FIXES END
"------------------------------------------------------------------------
------

"------------------------------------------------------------------------
------
"GENERAL CUSTOMIZATIONS
"
"DEFAULT-COLORSCHEME
" colorscheme neuromancer
" colorscheme ChocolateLiquor
" colorscheme made_of_code-dark
" colorscheme radicalgoodspeed
colorscheme genericdc




"START - Modifications for nerdcommenter plugin

" Add spaces after comment delimiters by default
let g:NERDSpaceDelims = 1

" Use compact syntax for prettified multi-line comments
let g:NERDCompactSexyComs = 1

" Align line-wise comment delimiters flush left instead of following
code indentation
let g:NERDDefaultAlign = 'left'

" Set a language to use its alternate delimiters by default
let g:NERDAltDelims_java = 1

" Add your own custom formats or override the defaults
let g:NERDCustomDelimiters = { 'c': { 'left': '/**','right':
```

```
'*/'  }  }

" Allow commenting and inverting empty lines (useful when commenting a
region)
let g:NERDCommentEmptyLines = 1

" Enable trimming of trailing whitespace when uncommenting
let g:NERDTrimTrailingWhitespace = 1

" Enable NERDCommenterToggle to check all selected lines is commented
or not
let g:NERDToggleCheckAllLines = 1

"END - Modifications for nerdcommenter plugin

"Set linewrap so that it doesn't split words.
set wrap linebreak nolist

"(1) Numbered Lines - Auto relative numbers
set number relativenumber

augroup numbertoggle
    autocmd!
    autocmd BufEnter,FocusGained,InsertLeave * set relativenumber
    autocmd BufLeave,FocusLost,InsertEnter   * set norelativenumber
augroup END

"(2) Auto indent on
set autoindent

"(3) Activate Syntax Highlighting.
" if !exists("g:syntax_on")
    " syntax enable
" endif
syntax on

" Allow Mouse
set mouse=a

"(4) Jump between tags using '%' *note if on the bracket it will just
jump to
"the opposite bracket*

"(5) Turn on matchit plugin (jumps between tags)
packadd! matchit

"GENERAL CUSTOMIZATIONS END
"------------------------------------------------------------------
------
```