The most powerful IDE

Lars Hübner
14. Mai 2018

DHBW
Duale Hochschule
Baden-Württemberg

Agenda

Introduction

Mode Concept

Basic editing

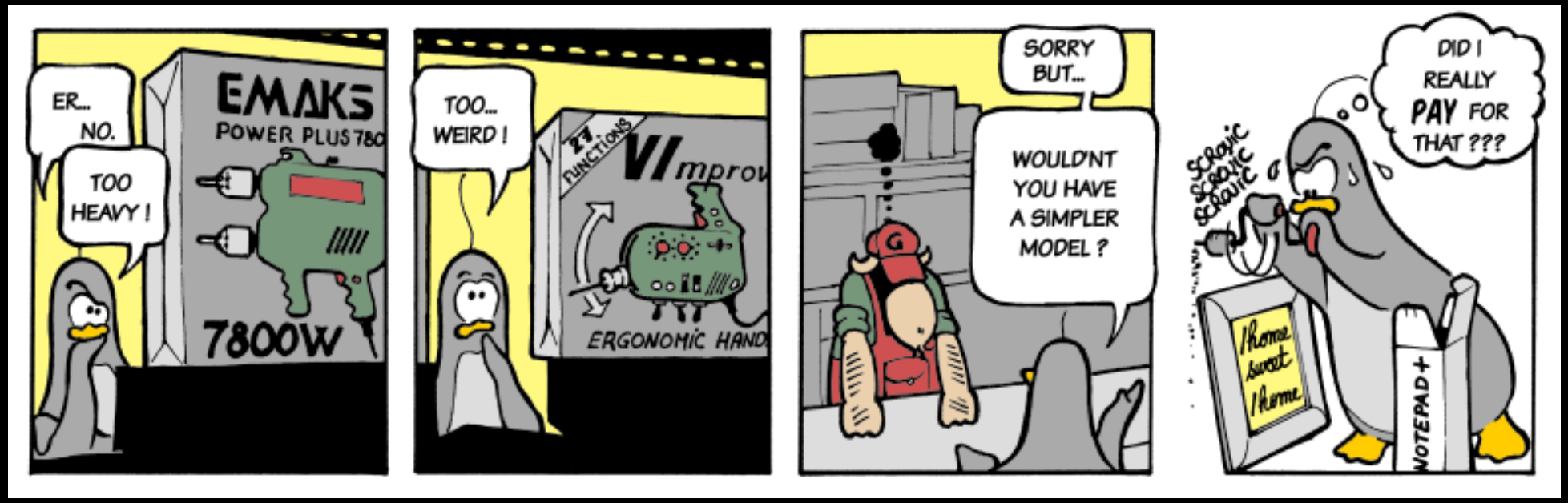
More interesting stuff

+

Plugins

Hands on part

What is Vim? (or Vi IMproved)



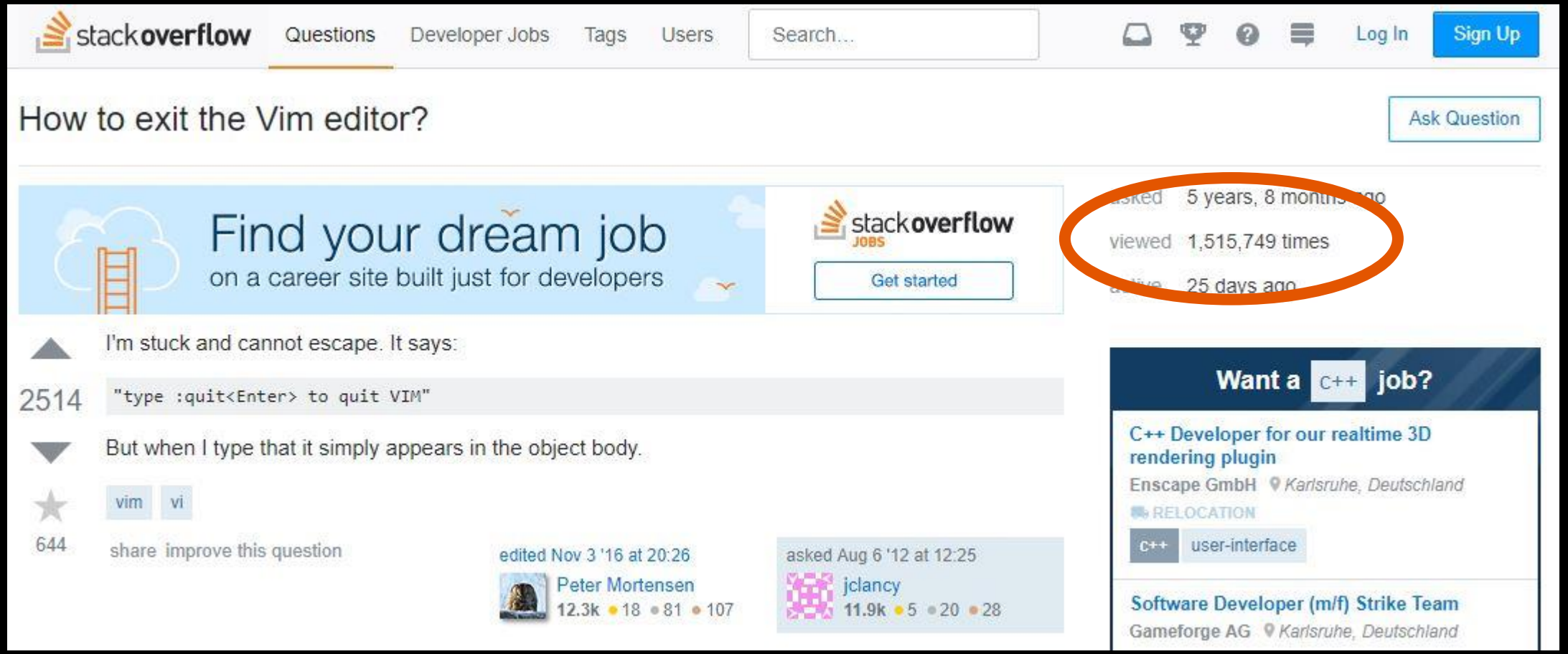
“Vim is a highly configurable text editor built to enable efficient text editing.”

“Vim isn't an editor designed to hold its users' hands.

It is a tool, the use of which must be learned.”

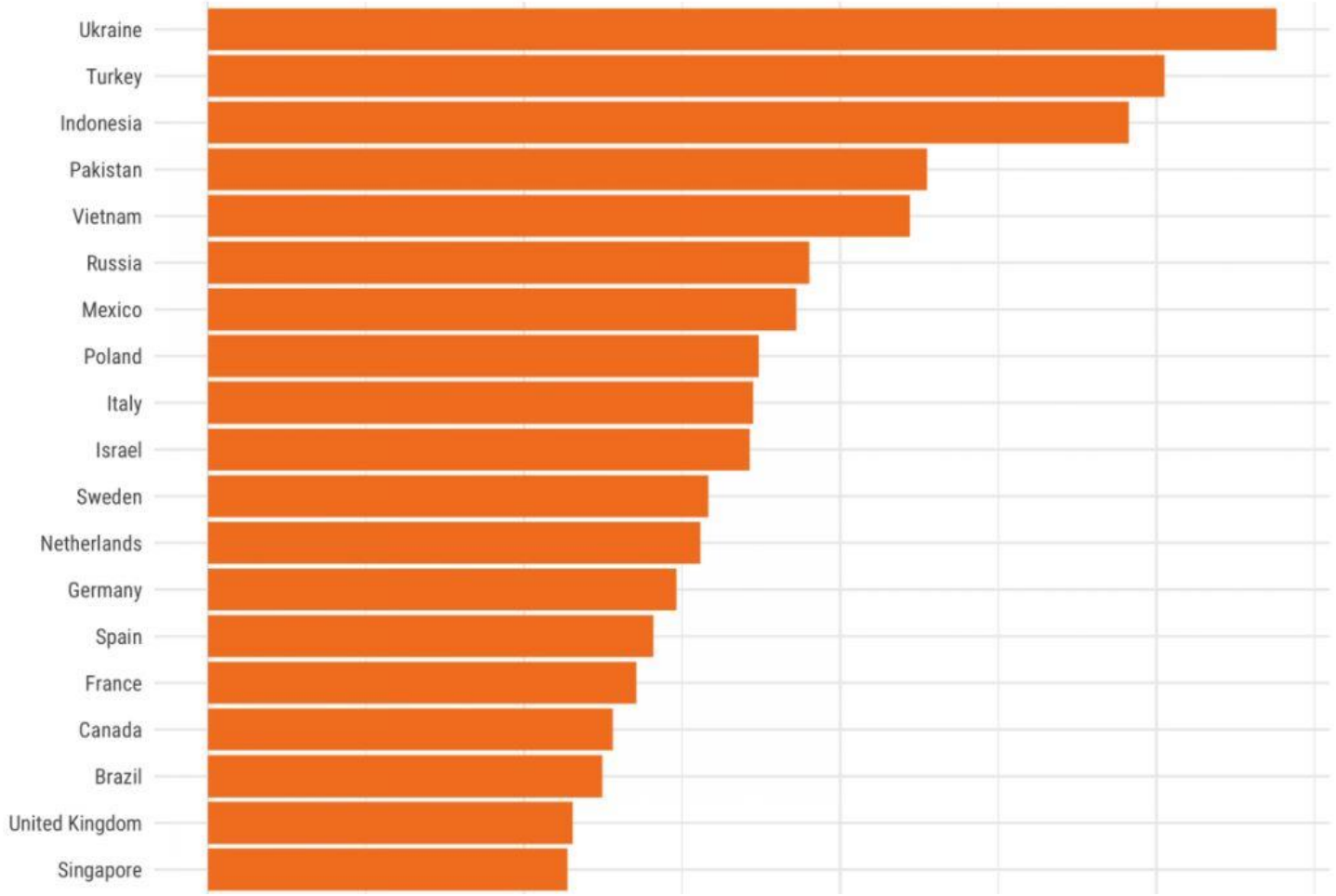
www.vim.org

What does the world think about Vim?



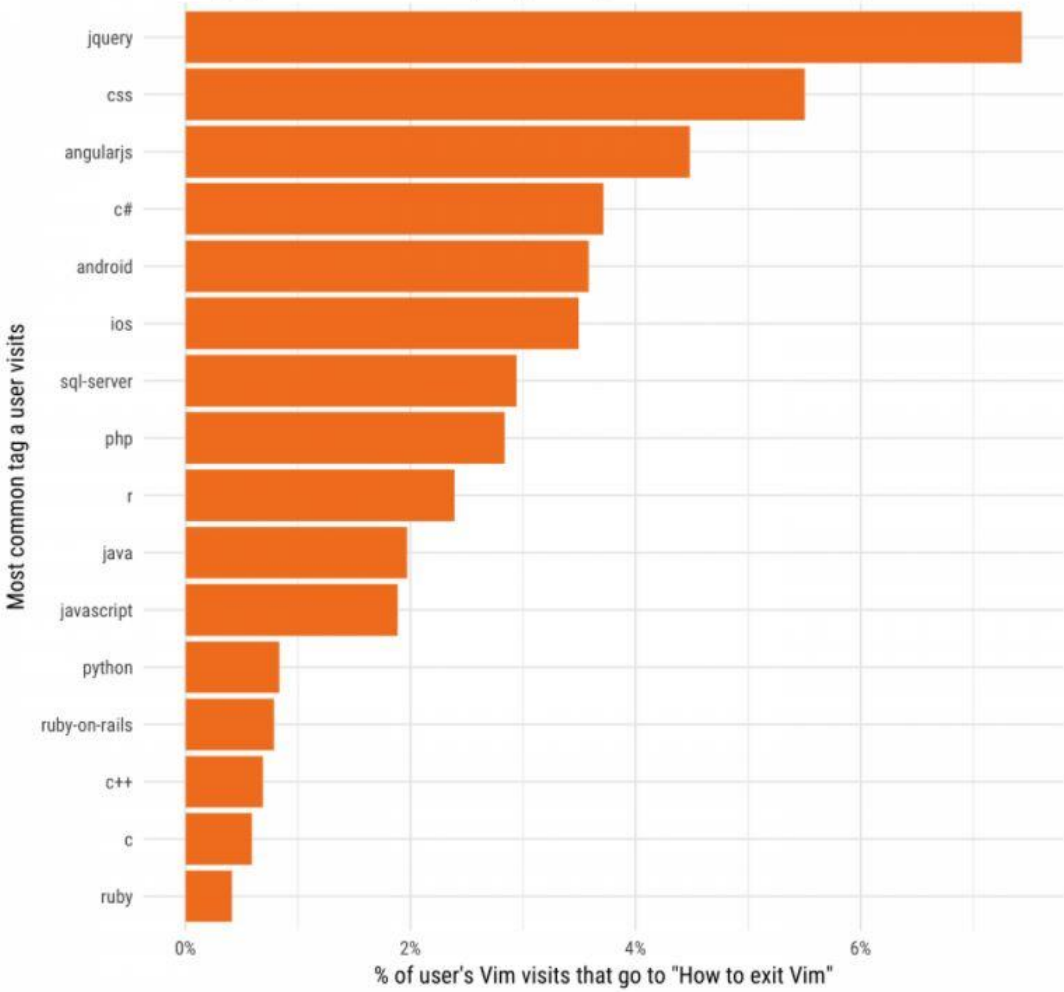
What countries are getting stuck in Vim?

Measured by a % of all Vim visits in one year of traffic. Only countries with >100 million visits in that year.



What developers are most likely to get stuck in Vim?

Divided by tag the user visited most frequently, showing the 16 most common such tags.



3534

Hit the Esc key to enter "Normal mode". Then you can type `:` to enter "Command-line mode". A colon (`:`) will appear at the bottom of the screen and you can type in one of the following commands. To execute a command, press the Enter key.

- `:q` to quit (short for `:quit`)
- `:q!` to quit without saving (short for `:quit!`)
- `:wq` to write and quit
- `:wq!` to write and quit even if file has only read permission (if file does not have write permission: force write)
- `:x` to write and quit (similar to `:wq` , but only write if there are changes)
- `:exit` to write and exit (same as `:x`)
- `:qa` to quit all (short for `:quitall`)
- `:cq` to quit without saving and make Vim return non-zero error (i.e. exit with error)

You can also exit Vim directly from "Command mode" by typing `ZZ` to save and quit (same as `:x`) or `ZQ` to just quit (same as `:q!`). (Note that case is important here. `ZZ` and `zz` do not mean the same thing.)

Vim has extensive help - that you can access with the `:help` command - where you can find answers to all your questions and a tutorial for beginners.

But who the hell had this idea?

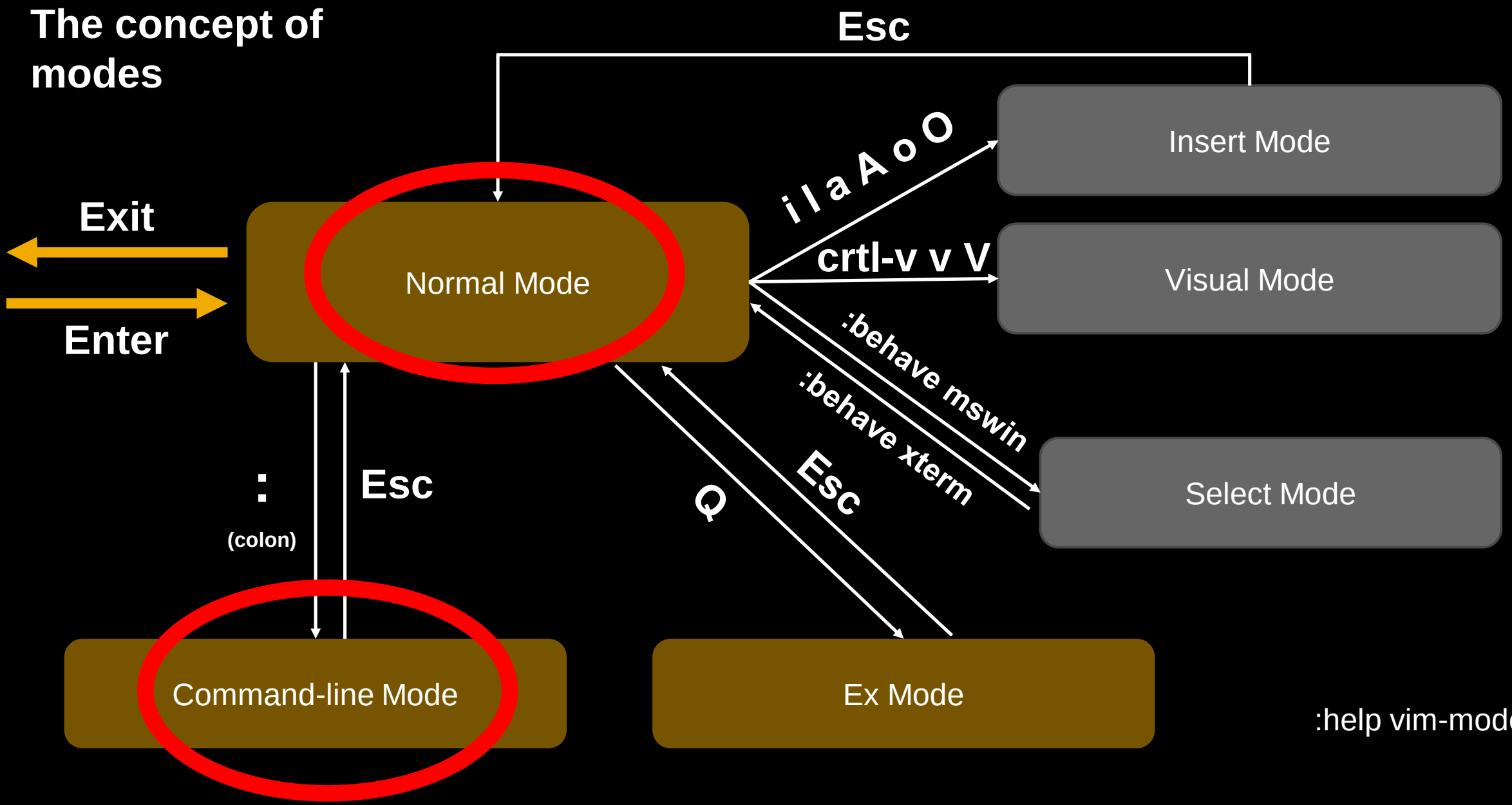
Bill Joy created Vi

1976

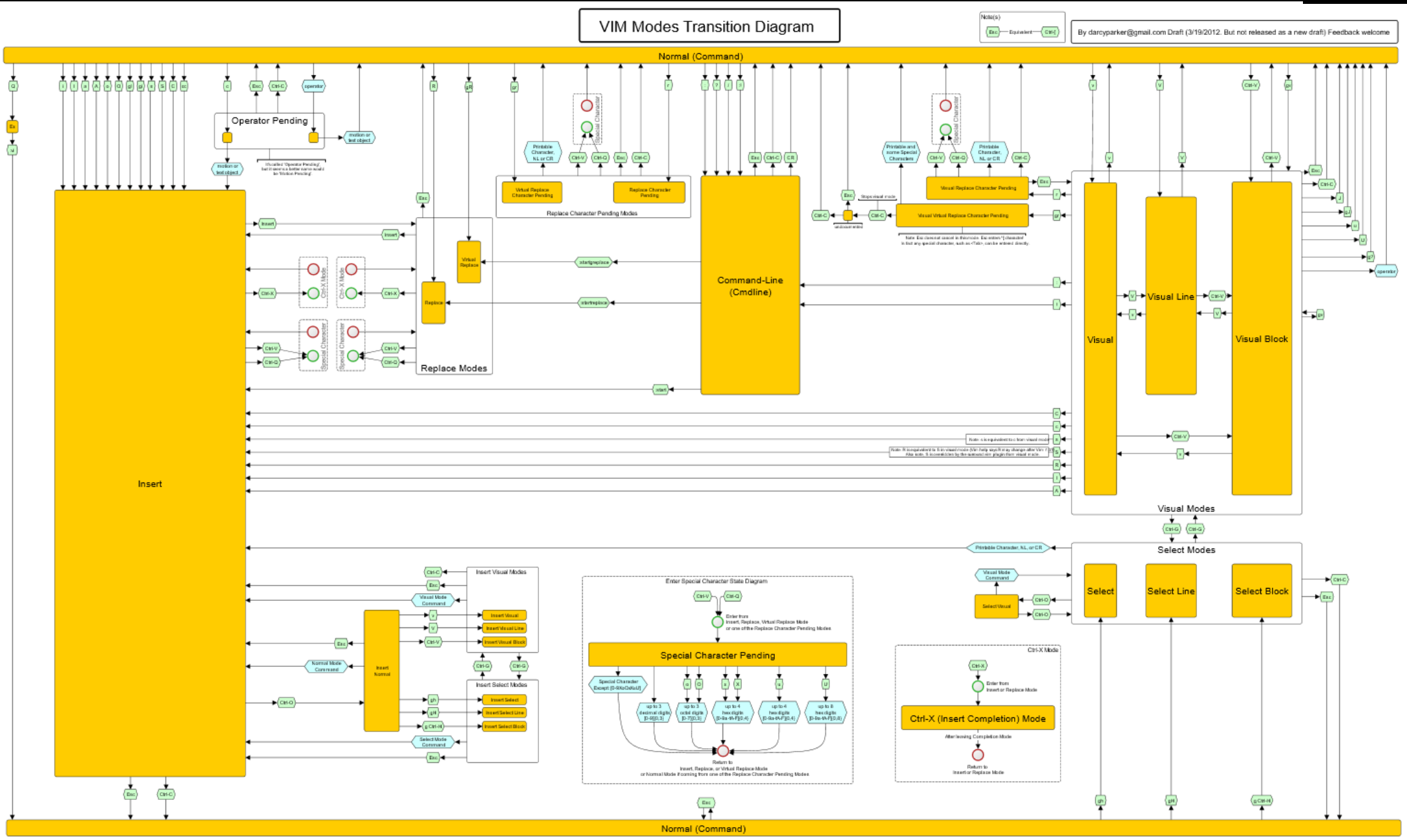
Bram Moolenaar created Vim

1991

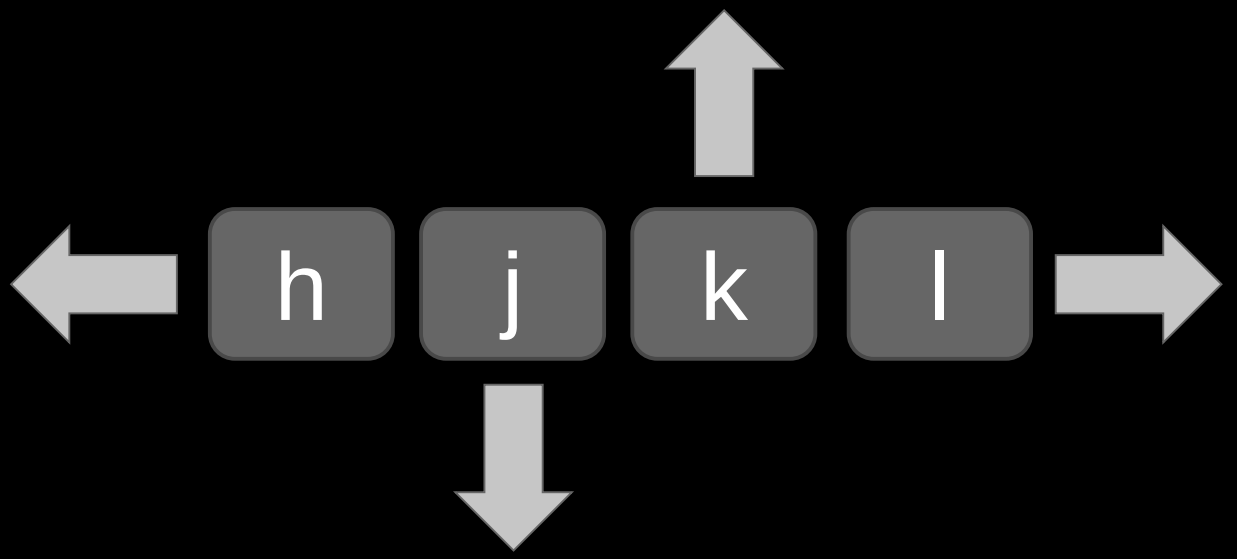
The concept of modes



:help vim-modes


VIM Modes Transition Diagram
Note(s)
Esc Equivalent Ctrl-[
By darcyparker@gmail.com Draft (3/19/2012. But not released as a new draft) Feedback welcome
Normal (Command)
Operator Pending
motion or text object
Insert
Replace Modes
Virtual Replace
Replace
Ctrl-X Mode
Special Character
Printable Character, NL or CR
Virtual Replace Character Pending
Replace Character Pending
Replace Character Pending Modes
Command-Line (Cmdline)
Visual
Visual Line
Visual Block
Visual Modes
Select
Select Line
Select Block
Select Modes
Insert Visual Modes
Insert Visual
Insert Visual Line
Insert Visual Block
Insert Select Modes
Insert Select
Insert Select Line
Insert Select Block
Insert Normal
Enter Special Character State Diagram
Special Character Pending
Ctrl-X Mode
Ctrl-X (Insert Completion) Mode
Normal (Command)

In Normal Mode



w	- to the beginning of the next word
e	- to the end of the word
b	- to the beginning of the previous word
0	- beginning of the Line
$	- end of the Line
1G	- first Line
#G	- Line Number #
G	- Last Line

[changing-command] [number / option] move-command

d - Delete

y - Yank

c - change

i - inner

a - all

p P - paste

Vim is a programmers editor

: 5, 10 fo - fold from line 5 to 10

za - toggle fold

>> << - indent line

=i{ - auto indention of the inner {…} part

% - jump to corresponding brackets

* # - find symbol under curser

gd - goto definition (of variable etc.)

/sometext - search for some Text

:%s/old/new/gc - replace global in this file with confirmation

:find filename - fuzzy finder for files

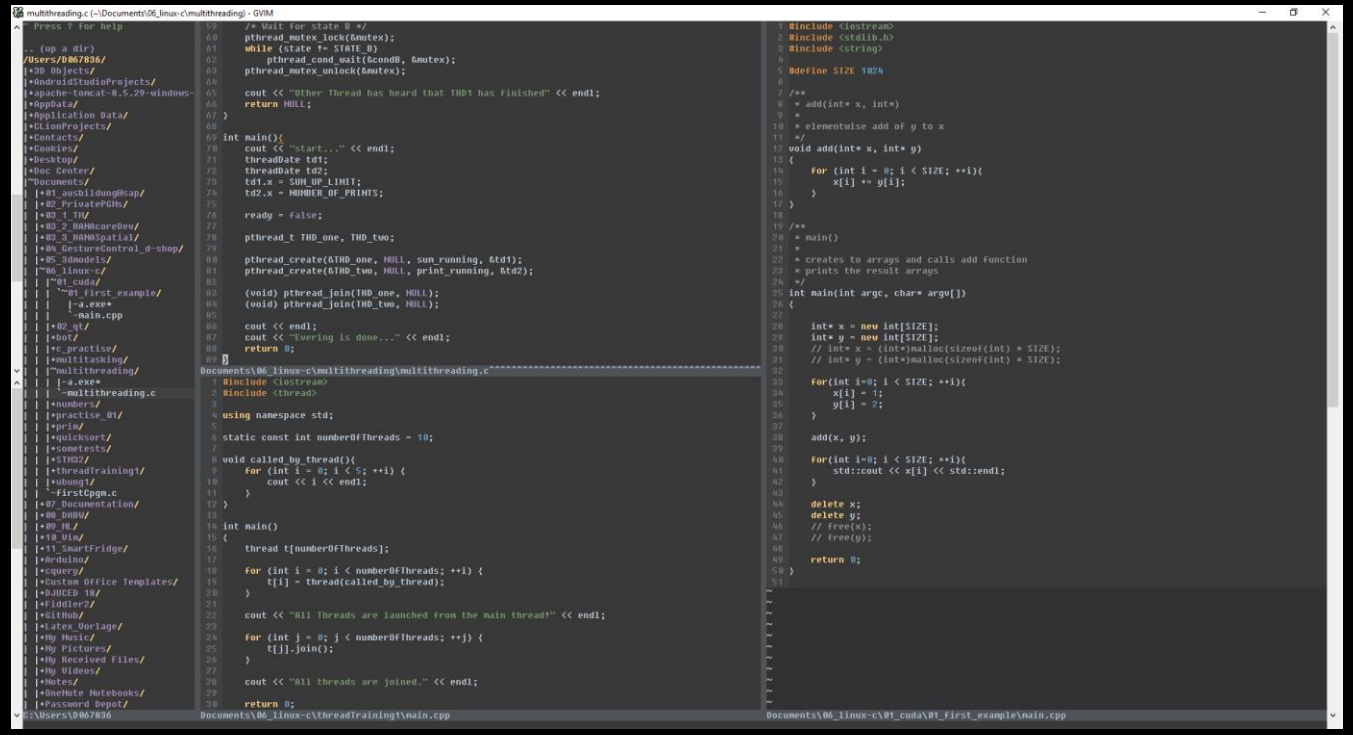
ctags

[Generates index file of names found in source code]

<C-]>	goto definition (push on stack)
:stag <C-W]>	show definition in split
g]	show definitions
<C-t>	pop from tag stack
:tags	show list of used tags

Handle multiple files in windows

Ctrl-w v	- vertical split
Ctrl-w s	- horizontal split
Ctrl-w h / j / k / l	- move between splits
Ctrl-w H / J / K / L	- move splits
Ctrl-q	- quit a split
:tabnew	- create new tab
gt	- go to next tab
gT	- go to previous tab
:Ex	- file tree
:e filename	- open file



You want to know where you are:

Ctrl-g

:help <whatever>

:help tutor

Let’s make Vim an IDE.

Folder Structure

```
d067836@WDFN33986757A:/mnt/c/Program Files (x86)/Vim$ tree -L 1
.
├── vim80
├── vimfiles
└── _vimrc
```

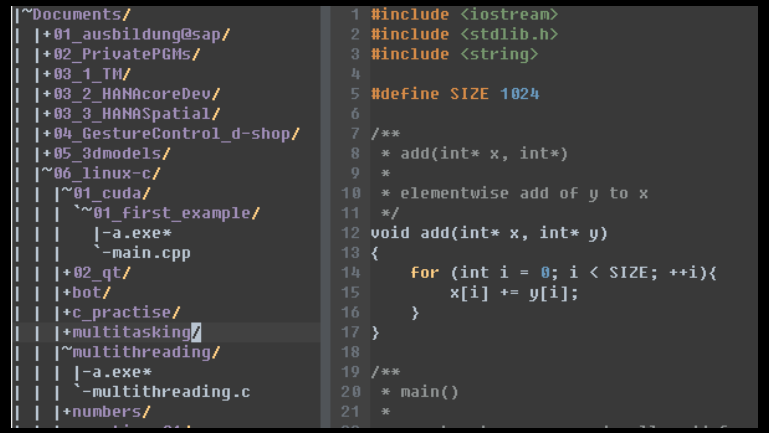
Vim Version 8.0 ➔ vim80 folder contains the tools

```
dr-xr-xr-x 2 root root 0 Apr 17 18:16 autoload
dr-xr-xr-x 2 root root 0 Apr 23 09:26 bundle
dr-xr-xr-x 2 root root 0 Apr 17 16:22 colors
dr-xr-xr-x 2 root root 0 Apr 17 16:22 compiler
dr-xr-xr-x 2 root root 0 Apr 25 12:04 doc
dr-xr-xr-x 2 root root 0 Apr 17 16:22 ftdetect
dr-xr-xr-x 2 root root 0 Apr 17 16:22 ftplugin
dr-xr-xr-x 2 root root 0 Apr 17 16:22 indent
dr-xr-xr-x 2 root root 0 Apr 17 16:22 keymap
dr-xr-xr-x 2 root root 0 Apr 17 16:22 plugin
dr-xr-xr-x 2 root root 0 Apr 17 16:22 syntax
```

vimfiles folder contains all the plugins and addons

_vimrc file contains all your settings and modifications

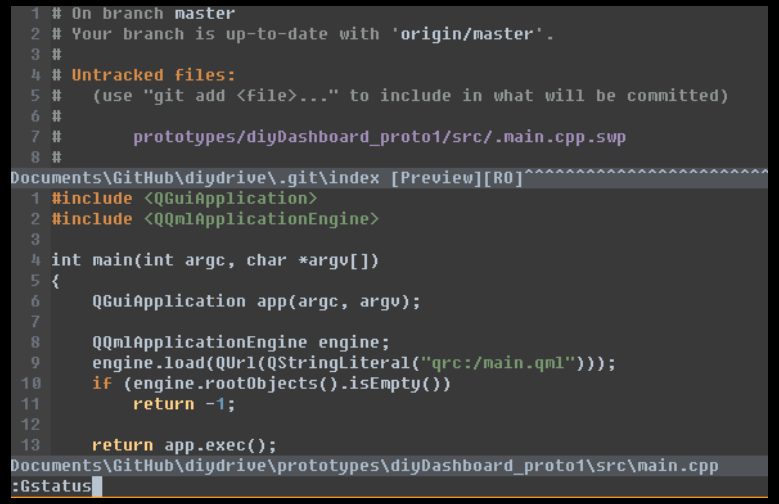
Let’s talk about Plugins…



NERDTree

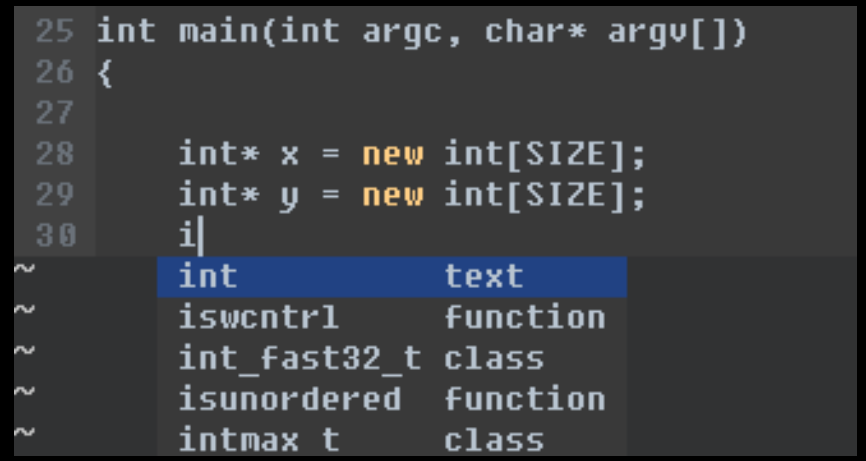
A more advanced

directory tree



Vim-fugitive

Git integration



Vim-lsp and the language server protocol

Autocompletion etc.

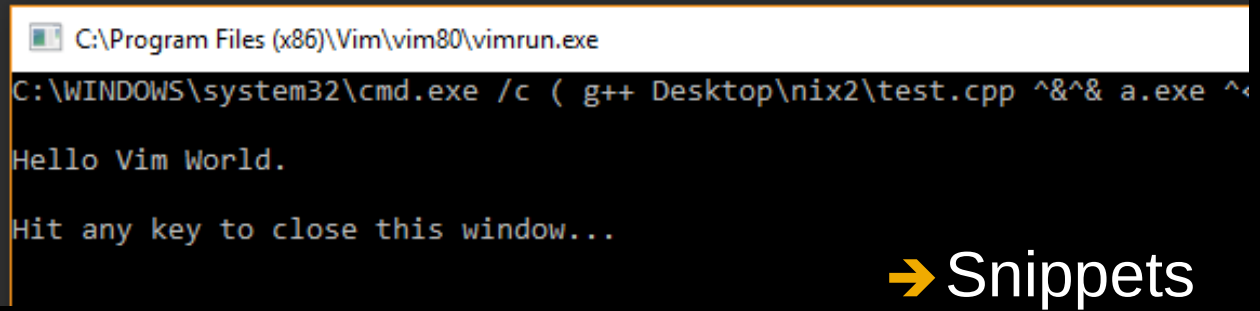
Some extras

```cpp
#include <iostream>

/**
 * main function
 */
int main(int argc, char* argv[]){

    std::cout << "Hello Vim World.\n" << std::endl;

    return 0;
}
```



- ➔ Snippets
- ➔ Compile and run with one key press

Why you should use vim?

gvim -d pgm1cpp pgm2.cpp

record with q<letter> and apply recording with @<letter>

:earlier 21m

…or use vim in intellij (_ideavimrc)

Thank you and happy vimming.

Contact information:

Lars Hübner

DHBW
Duale Hochschule
Baden-Württemberg

Autocompletion in vim

Language server protocol

- Install Language server
  - pip install python-language-server

- Install vim language server client
  - cd ~/path/to/vimfiles/bundle
  - git clone https://github.com/prabirshrestha/async.vim.git
  - git clone https://github.com/prabirshrestha/vim-lsp.git

  git clone https://github.com/vim-scripts/AutoComplPop.git

```
if executable('pyls')
    " pip install python-language-server
    au User lsp_setup call lsp#register_server({
        \ 'name': 'pyls',
        \ 'cmd': {server_info->['pyls']},
        \ 'whitelist': ['python'],
        \ })
    autocmd Filetype python setlocal omnifunc=lsp#complete
endif
```

C/C++ language server

git clone https://github.com/cquery-project/cquery --single-branch --depth=1

Install https://www.visualstudio.com/downloads/#build-tools-for-visual-studio-2017

Install python 3

Install 7zip

cd cquery

git submodule update –init

python waf configure build

add \cquery\build\release\bin to environmental variables

Sources:

https://upload.wikimedia.org/wikipedia/commons/9/9f/Vimlogo.svg

https://rawgit.com/darcyparker/1886716/raw/eab57dfe784f016085251771d65a75a471ca22d4/vimModeStateDiagram.svg

https://stackoverflow.blog/2017/05/23/stack-overflow-helping-one-million-developers-exit-vim/

https://de.wikipedia.org/wiki/Bill_Joy

https://en.wikipedia.org/wiki/Bram_Moolenaar

https://www.karlsruhe.dhbw.de