vim

"Who needs a mouse?"

Nicolas Dorrmann

04. Nov. 2020

History and Design Philosophy

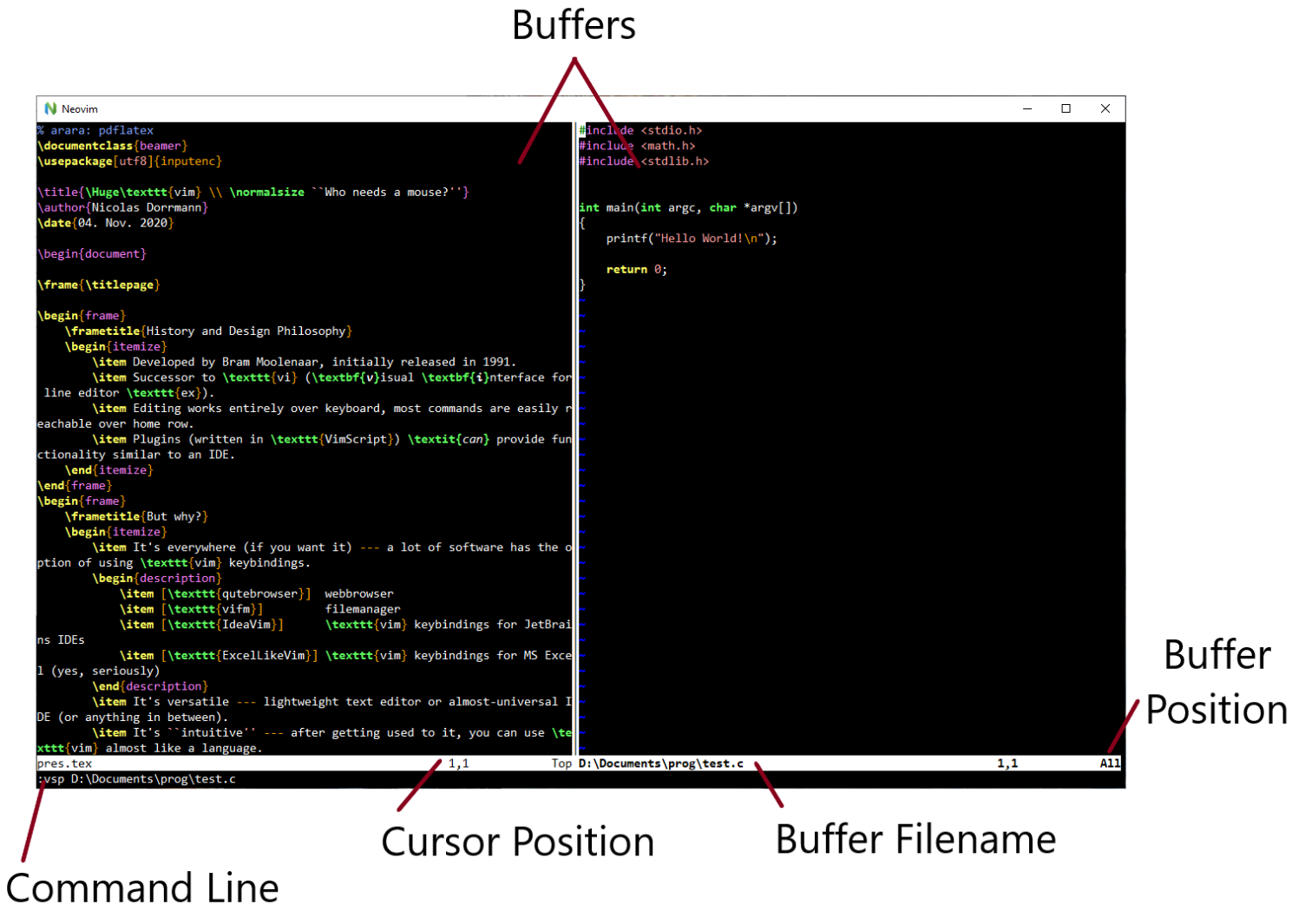
- Developed by Bram Moolenaar, initially released in 1991.
- Successor to `vi` (**v**isual **i**nterface for line editor `ex`).
- Editing works entirely over keyboard, most commands are easily reachable over home row.
- Plugins (written in `VimScript`) *can* provide functionality similar to an IDE.

But why?

- It's everywhere (if you want it) — a lot of software has the option of using `vim` keybindings.

  `qutebrowser` webbrowser
  `vifm` filemanager
  `IdeaVim` `vim` keybindings for JetBrains IDEs
  `ExcelLikeVim` `vim` keybindings for MS Excel (yes, seriously)
- It's versatile — lightweight text editor or almost-universal IDE (or anything in between).
- It's "intuitive" — after getting used to it, you can use `vim` almost like a language.

Basics - The Interface



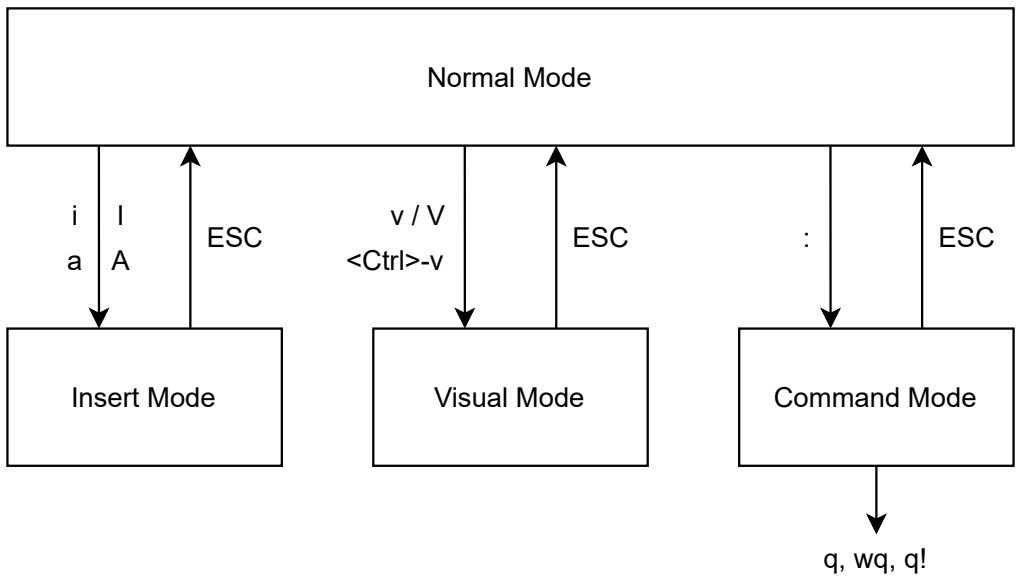
Basics - Modes

`NORMAL` navigate within the file, issue text commands (yank/change/delete)

`COMMAND` open, close, or save a file

`INSERT` direct input (i. e. actually typing text)

`VISUAL` text highlighting, text blocking



Basics

- Actions ("verbs")

yank	copy a movement
change	replace object with a new object
delete	remove an object

- Modifiers

NUM	number of objects to perform action on
i	perform action **i**nside an object
a	perform action **a**round an object
t	perform action up **t**o an object
f	perform action up to and including an object

- Text Objects ("nouns")

word	string of non-whitespace characters surrounded by whitespace
sentence	ends with `!`, `?`, `.` followed by newline
paragraph	consists of several sentences, delimited by empty lines

Commands

- Basic commands have the form `Action` — `Modifier` — `Object`
- Examples

  `y2w` copy the next two words

  `dis` delete the current sentence (i. e. the sentence the cursor is in)

  `cp` change (i. e. delete and enter `INSERT` mode) the current paragraph

Search and Replace

- Search
  - Forward search with `/string`. Backward search with `?string`.
  - Next result with `n`, previous with `N`.
- Search and Replace (`substitute`)
  - Basic syntax: `:[range]s/search/replace/`
  - `[range]` is a line, range of lines (`start, end`), or entire buffer (`%`).
  - Global replace with `g` or with confirmation (`gc`).

Regular Expressions

`.`	match one of any character
`[range]`	matches any one of the characters in `range`
`*`	match arbitrary many repetitions of preceding character
`\+`	match one or more of the preceding character
`\=`	match none or one of the preceding character
`{m,n}`	between `m` and `n` repetitions
`^`	match start of a line
`$`	match end of a line

`.vimrc` / `init.vim`

- Configuration file for `vim` settings.
- Written in `VimScript`, an interpreted programming language.
- Read and executed at editor startup (provided it exists).
- Allows various customization options:
  - font, syntax highlighting and color scheme
  - define or redefine keybindings
  - modify various preset options

VimScript Basics

`(i/c/v/n)map` (re)map a key combination

`set` (re)define an option

`let` (re)define a variable

`source` "include" an additional `.vim` file

`:help xyz` open documentation on `xyz`

Plugins and Plugin Managers

- http://vimawesome.com/ — An excellent resource for `vim` plugins.
- Plugin Manager
  - `pathogen.vim` – minimalist, simple to use, requires manual installation/deinstallation
  - `vundle` – requires listing all plugins in `.vimrc`
  - `vim-plug` – similar to `vundle`

Tutorial: Setting up a Productive `Neovim` Environment

In this tutorial, we will do the following:

1. Configure the `pathogen.vim` plugin manager.
2. Install the `Dracula` color scheme and `UltiSnips` snippet manager.
3. Create a few snippets for quickly creating Java classes.