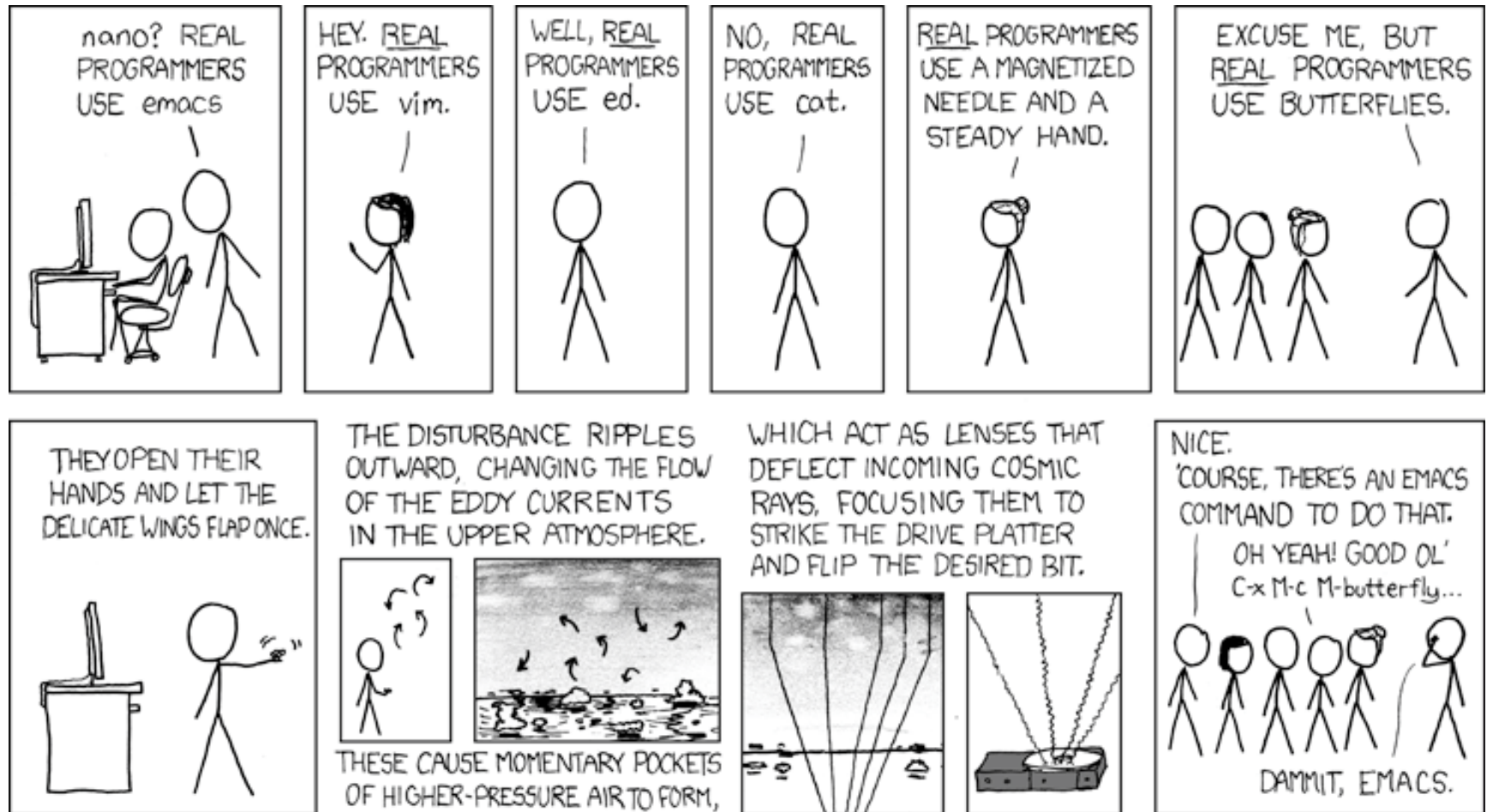


Text Editors for Programmers

EECS-678 Staff

Real Programmers



vim

- Based on vi
 - vi was written in 1976 and has become standard on Unix machines
- Basic design principles:
 - Retains each permutation of typed keys to resolve commands
 - Smaller and faster editor – but with less capacity for customization
 - Uses distinct editing “modes”

Using Vim on a Simple Example

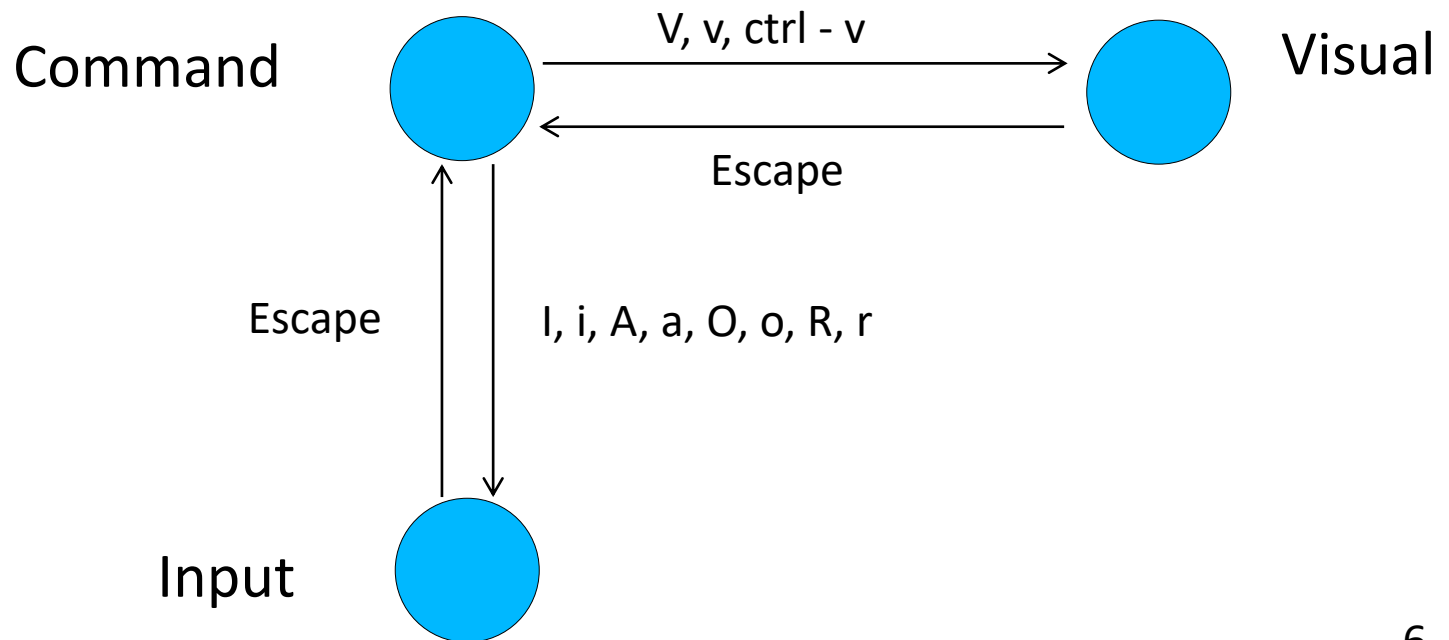
- You should have received two source files (simple.c and simple.h), a Makefile and a dot_vimrc file from the lab website
 - Save *dot_vimrc* as *.vimrc* in your home directory i.e.
 - `mv dot_vimrc ~/.vimrc`
- dot_vimrc
 - A collection of vim commands run each time you start vim
 - Used to set mappings / options that are not otherwise set by default

Using Vim to Create / Edit a File

- Start a session
 - `vim simple.c`
- Press 'i' to enter insert mode
 - Now type any text you want to insert
- Press 'Esc' to enter command mode
 - Type `:wq` to save changes and exit the session

Vim – Modes of Operation

- Command Mode
- Input Mode
- Visual Mode

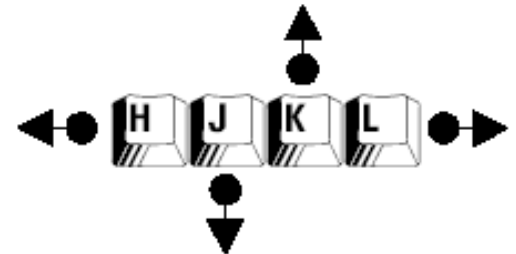


Essential Commands

- `:e <file-name>`
 - Edit *file* in a new buffer
- `:W`
 - Save any modifications to the current buffer
- `:q`
 - Quit Vim. If you have any modifications that you do not want to save, use `:q!`
- `u, <c-r>`
 - Undo, redo

Command Mode: Navigation

- Reopen simple.c
 - Use j, k, l, and h to navigate around the file as shown. This may take awhile to get used to, but is very nice once you have it down
 - For faster page scrolling, use <c-b> and <c-f> for page up and page down
 - These commands have been mapped to spacebar and backspace in the provided .vimrc script



Input Mode

- The following commands switch to input mode:
 - i – characters inserted just before the cursor position
 - I – characters inserted at the beginning of the line
 - a – characters inserted just after the cursor position
 - A – characters appended to the end of the line
 - o – characters inserted in a new line below the cursor
 - O – characters inserted in a new line above the cursor
 - C – Often overlooked, deletes the line after the cursor position and start inserting characters at this position
- After you're done editing, press Escape to go back to command mode, and :w to write the changes

Common Editor Commands

- Cut/copy/paste in command mode:
 - dd – cut a line of text
 - yy – copy (“yank”) a line of text
 - P/p – paste a line of text above / below the cursor position
- Commands in Vim can be applied to multiple lines by typing the number of lines you want before the command:
 - “12dd” cuts 12 lines of text
 - “4j” moves the cursor down 4 lines

Common Editor Commands

- Format a block of code to comply with text-width setting
 - `gq <motion command>`
 - `<motion command>` is any of the commands to move the cursor (i.e. `j`, `k`, `h`, and `l`)
- Format a block of code to conform to tabbing conventions
 - `==`
- See examples in `simple.c`

Searching

- Search for next occurrence of *word*
 - /word
 - jump to the next / previous occurrence of word
 - n / N
- Ignore case while searching
 - :set ic
- th – toggle search highlighting

Find / Replace

- *:s /search_for/replace_with/*
- Takes regex for strings to find / replace
- Variations
 - *:s /s/r/g* – Replace every occurrence on the line (not just the first)
 - *:%s /s/r/g* – Replace every occurrence in the current buffer
 - *:s /s/r/g 12* – Replace for the next 12 lines
 - *:s /s/r/gc* – Replace but get confirmation before doing so
 - *:s /s/r/gi* – Ignore case when searching for s

Setting the Mark

- `ma` – Sets the mark `a` to the current cursor position
 - `a` is not unique, any alphanumeric character can be used
- Now, pressing ``a` in command mode returns you to the position marked by `a`
 - Helpful for getting back to hard to find sections of code quickly
 - See the example in `simple.c` that shows how it can be used with the `find/replace` command to comment out large sections of code
- `da` – Deletes the mark `a`

Visual Mode

- V/v enter into visual mode
- Allows user to visually select text for commands
- Navigate in visual mode as in command mode (g,j,h,k)
- Issue commands with selected text ('y' to yank, 'd' to cut, etc.)
- Escape exits visual mode

Buffers

- Vim allows you to edit multiple files in one session using buffers
 - `<c-w> v` to split the screen vertically
 - `<c-w> s` to split the screen horizontally
 - `<c-w> w` to switch to the other screen
 - `:S / Vex` – splits the screen horizontally or vertically and opens a file explorer in the new screen
- Select `simple.h` to open it in the new screen

Tagging the Source

- Big advantage to Vim is its integration with a source code tagging program
- Inside a terminal, goto the directory of the simple source and type:
 - `ctags -R`
- Should create a file named tags. Now, reopen simple.c in Vim

Using Tags with Vim

- `<c-]>` - With your cursor over a variable, jump to the declaration of that variable
- `<c-t>` - Having jumped to a declaration, go back to the spot you jumped from
- You can use `<c-]>` multiple times before using `<c-t>`. The functionality operates like pushing and popping frames on a stack
- Extremely helpful for browsing and learning large programs

Colors

- Color-schemes can be downloaded from:
 - <http://www.cs.cmu.edu/~maverick/VimColorSchemeTest/>
- Current default color schemes for EECS machines are in:
 - `/usr/share/vim/vim72/colors/`
- Set a new color scheme with:
 - `:colorscheme name`

Vim Resources

- Vim Tips Wiki
 - http://vim.wikia.com/wiki/Main_Page
- Vim Cookbook
 - <http://www.oualline.com/vim-cook.html>
- Slashdot comments discussing Vim tips
 - <http://ask.slashdot.org/article.pl?sid=08/11/06/206213>
- For everything else, just use Google

TMUX

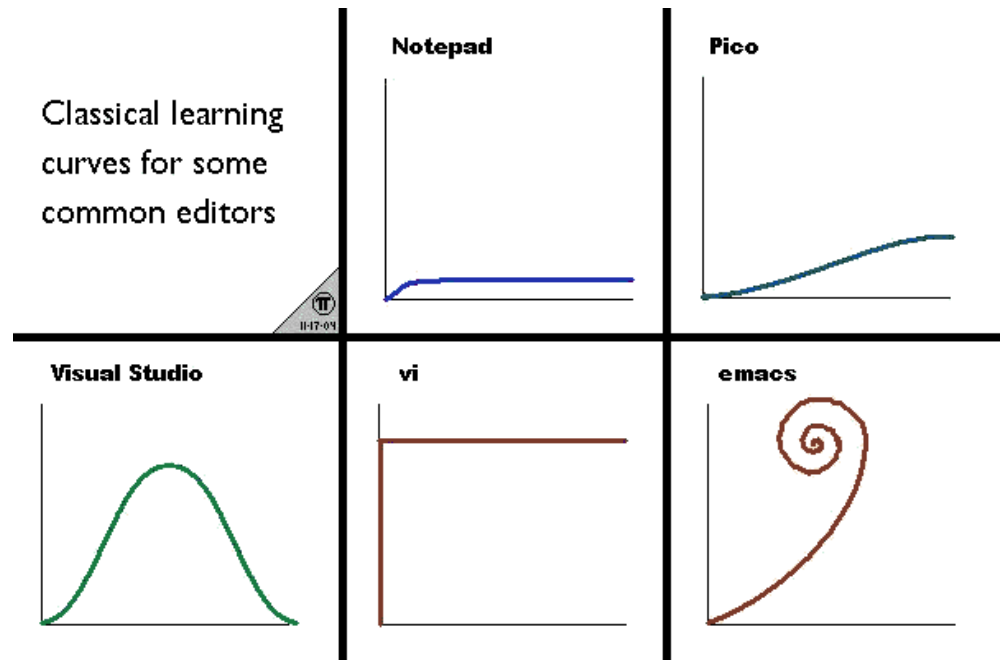
- tmux is a 'terminal multiplexer'
- Allows users to access multiple separate terminal sessions from within a single terminal window or remote terminal session

TMUX Essential Commands

- Commands outside tmux
 - Create a new session: `bash> tmux`
- Commands within tmux
 - Create a new terminal: `<c-b> + c`
 - Cycle through terminals: `<c-b> + n`, `<c-b> + p`
 - Kill terminal: `<c-d>`

Comparison of Editors

- Vim vs. Emacs
 - Vim is primarily an editor
 - Emacs is a Lisp interpreter running an editor
- Wikipedia: Editor War



Emacs Properties

- Single mode editor
 - Each key is a command to add the letter to the buffer
 - Key combos shortcut commonly used commands
- “C-x C-s” to save a file to disk
- “C-x C-c” to exit Emacs
 - Access all commands with “M-x <command name>”
- Highly extensible and versatile
 - Plugins for almost any functionality
 - Emacs Lisp (Elisp) programming language
 - More than just a text editor:
 - M-x org-mode, M-x shell, M-x life, M-x list-packages
- If you like Vim key-bindings install the evil-mode package via M-x list-packages or use the Spacemacs configuration

.emacs

- Similar to .vimrc
 - An Elisp script run at start up
 - Used for configuring Emacs options and attaching plugins
- Follow the instructions for the dot_vimrc file on the dot_emacs file while changing the file names where necessary

Emacs Navigation and Command Shortcuts

- Command shortcut syntax:
 - C- means hold Ctrl key
 - M- means hold Alt key or press Esc key
- Forward 1 character: C-f
- Backward 1 character: C-b
- Go to previous line: C-p
- Go to next line: C-n
- Forward 1 word: M-f
- Backward 1 word: M-b
- Page up: M-v
- Page down: C-v
- Go to top of buffer: M-<
- Go to bottom of buffer: M->
- Go to line “n”: M-g g “n”
- Repeat following command “n” times: <Esc> “n” command
- Repeat last command:
 - C-x z [z z z ...]
- All commands can be accessed with M-x “command name”

Basic Emacs Command Shortcuts

- Start Emacs from the command line:
 - emacs
 - emacs "file name"
- Close session: C-x C-c
- Suspend: C-x C-z
- Open file: C-x C-f
- Save file: C-x C-s
- Undo = Redo: C-_ or C-x u
- Abort command: C-g
- Begin mark region: C-<space>
- Copy region: M-w
- Kill region (Cut): C-w
- Yank (Paste): C-y
- Search buffer contents: C-s
- Select buffer: C-x C-b
- Previous buffer: C-x <left>
- Next buffer: C-x <right>
- Window - split:
 - Vertically: C-x 2
 - Horizontally: C-x 3
- Window - cycle cursor: C-x o