Using Vi(m)

- This presentation on using the Vi(m) editor will cover:
  - A brief overview of vim's features
  - Several demos on using vim

Using Vi(m)

- References:
  - Vimdoc: http://vimdoc.sourceforge.net/
  - http://www.vim.org/tips/index.php
  - http://www.vim.org/vimscriptlinks.php
  - Joe 'Zonker' Brockmeier has written several articles at Linux.com
    - http://www.linux.com/articles/#
    - 54157, 54159, 54936, 114138

Using Vi(m)

- Vi is a pocket knife, vim is a Swiss Army knife
- Probably even Bram Moolenaar doesn't use all of vim's features
- If you haven't edited a .vimrc file, you're not really using vim

Using Vi(m)
Basic Editing

- vi <filename> | <filelist>
- Two modes: command and input
  - Normal mode:
    - Move around file
    - Modify blocks of text
    - Manage file
    - Much, much more
  - Input mode
    - Enter text

Using Vi(m)
Basic Editing

- A session starts in normal mode
- To start entering text, press:
  - a: Start entering text after cursor
  - i: Start entering text at cursor
  - o\O: Start entering text on next\previous line
- To exit input mode, press ESC
- To exit vim:
  - :wq: Exit and save changes
  - :q!: Exit and discard changes

Using Vi(m)
Basic Editing

- Moving around
  - In Linux, arrows and page keys work as expected while standard keys are:
    - Ctrl-F\B: Scroll forward\backward 1 page
    - Ctrl-D\U: Scroll forward\backward 1/2 page
    - H\M\L: Top\middle\bottom of page
    - h\j\k\l: Move cursor left\down\up\right
    - w\b\e: Go to next\previous\end of word
  - #G: Go to line # (default = end of file)

Using Vi(m)
Basic Editing

- /\?<text>: Find text after\before current position
  - text = A string or any valid regular expression
    - ^\$ = From start\end of line
    - . = Any character
    - * = All of previous character (ie. a* all a's)
    - \ = Escape or treat special characters as regular characters
    - [] = Any character within brackets (ie. m[ae]n finds man or men)
    - \<\> = Target must be full word (ie. \<men\> will not find amen)

Using Vi(m)
Basic Editing

- Manipulate blocks of text
  - #CMDd
    - # = optional number of items
    - d = optional destination
      - 0\$: beginning\end of line
      - (\): beginning\end of sentence
      - {\}: beginning\end of paragraph
      - %: matching paren, bracket, or brace
      - w: word
    - CMD (general rule):
      - upper case = before current position
      - lower case = after current position

Using Vi(m)
Basic Editing

- Commands
  - d: Delete
  - y: Yank or copy
  - p: Paste
  - c: Change
- Some commands do not use destination:
  - x: Delete character
  - s: Substitute character

Using Vi(m)
Basic Editing

- Command line = :<command>
- Substitute
  - :b,e s/<find text>/<replace text>/g
    - b: Start substitution: line # or . (current line)
    - e: End substitution: line #, . , or $ (last line)
    - /: May be any non-alphanumeric character
    - g: Change every instance, blank = change first instance
  - find text: A valid regular expression
    - foo[bcj]ar finds foobar, foocar, and foojar
    - ^$ finds blank lines

Using Vi(m)
Basic Editing

- Saving and exiting
  - :w<filename> = Save current or named file
  - :q[!] = Quit [without saving changes]
- Editing multiple files
  - :#n = Edit next file in list (skip to #th file)
- Options
  - :set = See the documentation for details on set
- External commands
  - :!CMD such as :!ls -l *.c to list source code files

Using Vi(m) Editing Code

- Vim scripts
  - Should be installed in system vim directory, but may be installed in $HOME
  - Include scripts for vim tutoring, spell checking, color hilighting, syntax checking, even games
  - Most useful for editing code

Using Vi(m) Editing Code

- Enable syntax checking
  - :syntax enable
- Select alternate color scheme for syntax hilighting
  - :color <color scheme>
  - color schemes are in colors in vim system directory
- :filetype indent on
- :set tabstop=#

Using Vi(m) Editing Code

- Fold lines to hide them from view
  - #zf or zfd
    - d = same destination as other editing commands
  - May also enter v for visual block, move cursor to end of lines to fold and enter zf
  - Handy function key settings:
    - :map <F5> zf}
    - :map <F6> zf%
  - zo\c: Open\close fold
  - zE: Erase all folds

Using Vi(m) Editing Shortcuts

- Macros
  - :map KEY CMD
  - <Enter>, <Esc>, etc. may be used so those keys can be included in complex macros
  - Available keys for mapping:
    - <Fn> = Function key
    - <S-Fn> = Shift + Function key
    - <xFn> = Alt + Function key
    - ,LETTER_COMBINATION such as ,hi for html italics (Note: multi-character macro keys must be typed quickly or not recognized)

Using Vi(m)
Editing Shortcuts

- Autocmd: Define actions for certain events
- :au EVENT FILE_SPEC CMD
  - Events:
    - BufWinEnter: When a file is displayed in a window
    - BufWinLeave: Before a file is exited
    - BufNewFile: When starting to edit a new file
    - Many, many more
  - Commands: Apply generally to all files defined
    - For *.html, enter HTML, HEAD, and BODY tags
    - For all files, load and save status

Using Vi(m)
.vimrc

- Any command entered using “:” may be loaded automatically by entering it in $HOME/.vimrc
- Examples:
  - set history=200
  - set viewdir=~/.vimsess
  - au BufWinEnter * silent loadview
  - au BufWinLeave * mkview
  - map <F5> zf}
  - map <F6> zf%

Using Vi(m)
Conclusion

- Vim is a very powerful editor
- Learn the basics and then add knowledge as needed
- Learn regular expressions!!
- Learn to navigate vimdoc
- Experiment
- Share your experience with others