

# The Vicious Interface

Benjamin Brewster

Except as noted, all images copyrighted with Creative Commons licenses,  
with attributions given whenever available

# If It's So Vicious and Old, Why Study vi?

- History
  - Understanding why it exists and why it was created informs us about the underlying OS (UNIX) and the language it was developed in (C)
- Power
  - There are LOTS of things you can do in vi you can't do anywhere else
  - Important for manipulating large data files, repetitive commands, etc.
- Ubiquity
  - Installed on every UNIX and UNIX-like system!
- Necessity
  - Sometimes you'll have no other options because of the environment

# Text Editors

- There are many text editors available on UNIX
  - ed (a line editor only)
  - ex (an extended line editor; vi's predecessor)
  - emacs
- vi was written by Bill Joy in 1976 for BSD
  - Added full-screen visibility to ex
- Its name comes from the shortest unambiguous abbreviation of **visual**

# What's a line editor?



Image by user Chlor~enwiki, CC SA 3.0

“[ed is] the most user-hostile editor ever created.”

-Peter H. Salus, computer historian

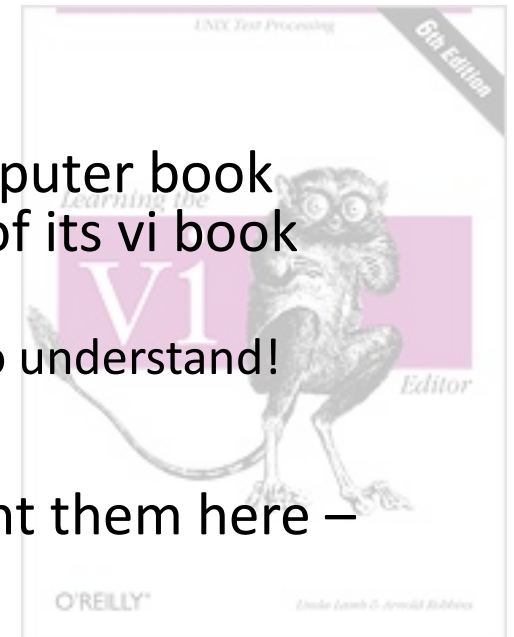
```
$ ed
a
ed is the standard Unix text editor.
This is line number two.
.
2i

.
,1
ed is the standard Unix text editor.$
$
This is line number two.$
3s/two/three/
,1
ed is the standard Unix text editor.$
$
This is line number three.$
w edfile
65
Q
$ cat edfile
ed is the standard Unix text editor.

This is line number three.
```

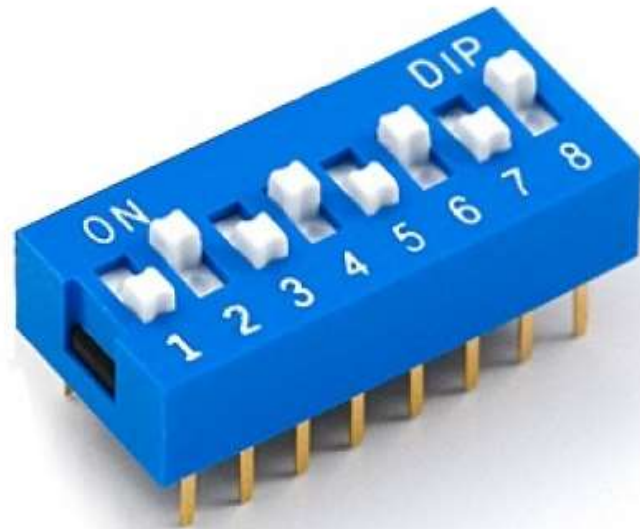
# vi = Vicious Interface

- Just because it's got a full-screen interface, it doesn't mean it's easy to use - but it is very powerful!
- In 1999, Tim O'Reilly, founder of the eponymous computer book publisher, stated that his company sold more copies of its vi book than its emacs book...
  - Not because more people *like* vi, but because it's harder to understand!
- Don't try to memorize all of the keystrokes as I present them here – just be aware they exist!



# Modes, modes, modes

- vi features one of the first visual environments, instead of line editors
- Primary paradigm: vi is modal
  - Normal mode
  - Insert mode
  - Command mode
  - Visual mode
  - and a few others



# A Visual View of vi

I can't find the tilde key  
This is a new line  
what?

# Cursor

## Blank lines

End of file

Mode

```
-- INSERT --
```

## Screen position in file

- Top == 0%
- Bot == 100%
- All == entire fire visible

## Cursor position

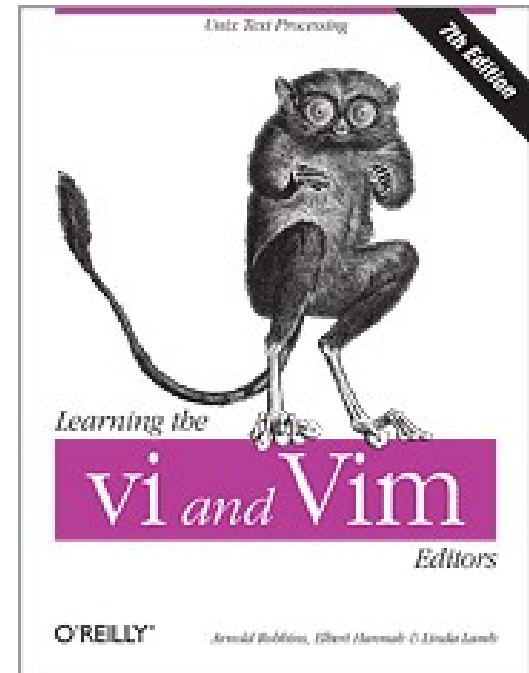
3, 6

All

# Actually, that was vim

- vim is **vi improved**
  - Better than vi, though the basic commands we're covering work in both
- vim is:
  - Still in development
  - vi is often mapped to simply start vim
- Starting vim

```
$ vim newFile
$ vim existingFile
```





# Modes for Real

## Normal

- Move around the document
- Perform one-shot edit commands on characters, paragraphs, or even larger blocks of text

## Insert

- Insert text into the document
- What normal WYSIWYG editors can only do



# Modes for real

## Replace

- Overwrite mode

## Visual

- Selects an area of text to which subsequent commands can be applied

## Command

- Whole file commands
  - Save, quit, search, etc.



# Normal Mode

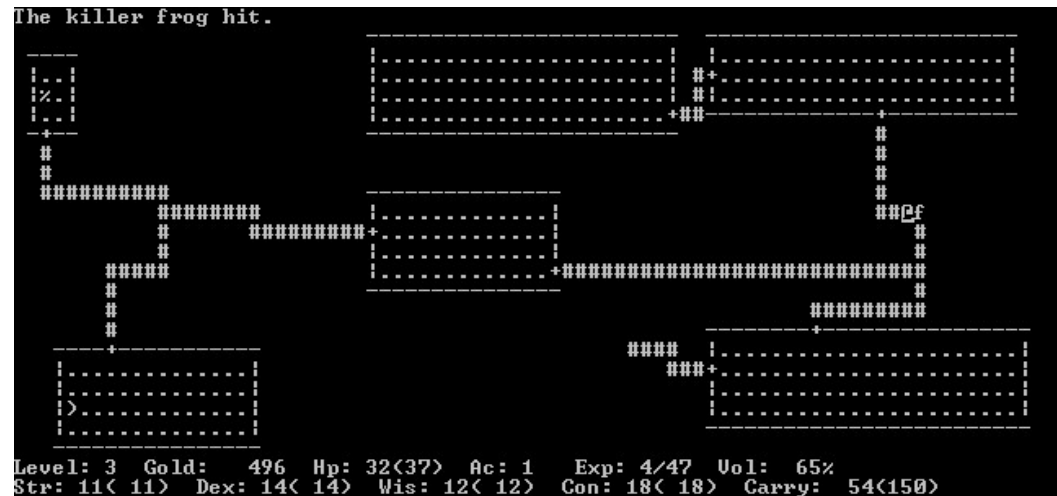
- Movement

- Cursor movement

- h, j, k, l - the Rogue keyset
    - \$ - move to the end of the line
    - 0 - move to the beginning of the line
    - w - move to beginning of next word to the right
    - b - move back to beginning of the previous word on the left

- Screen movement

- ^d - move screen down half a screen
    - ^u - move screen up half a screen



# Normal Mode

- How to get into Normal Mode
  - vi starts in Normal Mode
  - Hit escape
    - You can always hit escape – the key can never do anything but take you to Normal mode
    - In fact, hit it a bunch of times
    - Will beep if you're in Normal mode already



# Insert Mode

- Type like normal
- Move around with the arrow keys
  - Commands (including movement commands) from Normal mode will not work - you get characters instead



# Insert Mode

- How to get into insert mode
  - From Normal mode, hit **i**

i like catS In Normal Mode

hit **i**




i like catS Now in Insert Mode


type '2'

i like cat2S Still in Insert Mode

# Insert Mode

- In this situation, use a:

i like catS  In Normal Mode  
hit a (for append)  
i like cats  Now in Insert Mode  
hit 2  
i like cats2  Still in Insert Mode



# Replace



Like when you accidentally hit the Insert key in Word

- Overwrite mode
  - Non-insertion typing
- Two ways to get into Replace Mode
  - `r` – replace the character that the cursor is over with the next one typed
  - `R` – enter Replace Mode until we manually leave it (eg, by escape back to Normal Mode)



# Visual Mode

- Visual mode allows you to select text and then apply commands to it
- What you have selected is marked by **inverted** characters

# Visual Mode Demo

- Let's cut, copy, and paste:

myline

hit v

In Normal Mode

In Visual Mode

myline

hit 'l' three times

myline

hit y to yank (copy)

myline

hit p to paste

Note the insertion point for pasting is on the right

mylineine

# Command Mode



- Used to enter commands that pertain (mostly) to the entire file
- These commands are actually carried over from the line editor ed!
- To **save** your file, enter command mode (:), hit w, then enter:  
`:w`
- If you started vi without a filename, you'll have to type in a name and then hit enter:  
`:w myNewFileName`
- Can also be used to save a *copy*, but you'll still be editing the *original* file:  
`:w thesis_backupcopy`

# Quitting vi

- Quit:

`:q`

- Save, and then quit

`:wq`

- To exit without saving:

`:q!`

- From Normal Mode, you can save the current file and exit immediately:

`ZZ`



Note the lack of colon here

# Search and Replace

- To find a string pattern in the file:

`/pattern`

Note the lack of colon here

`n` will move you to the next instance of that `pattern`

`N` will move you to the previous instance of that `pattern`

- Remove highlighting after search:

`:nohl`

- Global search and replace:

`:%s/wrongtext/righttext/g`

# Advanced Command Mode

- Run a single UNIX command (from inside vi):

`: ! UNIXCOMMAND`

- Run a single UNIX command and insert the output into the file:

`:r ! UNIXCOMMAND`

- Put vi in the background, and start a new shell in the foreground (defaults to what is in your SHELL environment variable):

`:sh`



# Back to (Advanced) Normal Mode

- cut  
In visual mode, use `d` instead of `y`
- Delete/cut a line  
`dd`
- Copy the current line  
`yy`
- Undo the last Normal Mode command  
`u`



# Advanced Normal Mode

- Delete the current character

x

- Delete the current word

dw

- Transpose current and next char

xp

- Go into Append/Insert Mode at the end of the line

A





# Advanced Normal Mode

- Open new line above the current line in Insert Mode  
O (big oh)
- Open new line below the current line in Insert Mode  
o (little oh)
- Delete the rest of the line from where the cursor is  
d\$
- Delete the current char and enter Insert Mode  
s

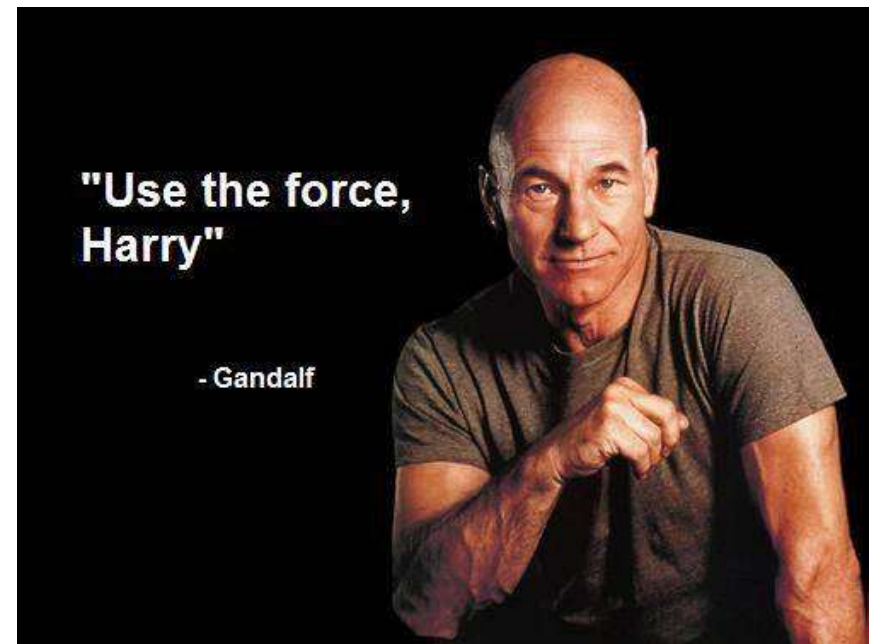
# Advanced Normal Mode

- Join two lines

```
firstlineALLONELINEfirstline
ALLONELINE
secondlineALLONELINEsecondli
neALLONELINE
~
```

Hit J

```
firstlineALLONELINEfirstline
ALLONELINEsecondlineALLONEL
INEsecondlineALLONELINE
~
```



# Advanced Normal Mode

- Delete the five lines starting at the current cursor position

`5dd`

- You can find further goofiness online on the “vi Resources” page
  - Like ~ which switches the case of a letter

