

LIS-3353

Linux/Unix Command Line Goodness

For comparison – Win 7?



OS X?



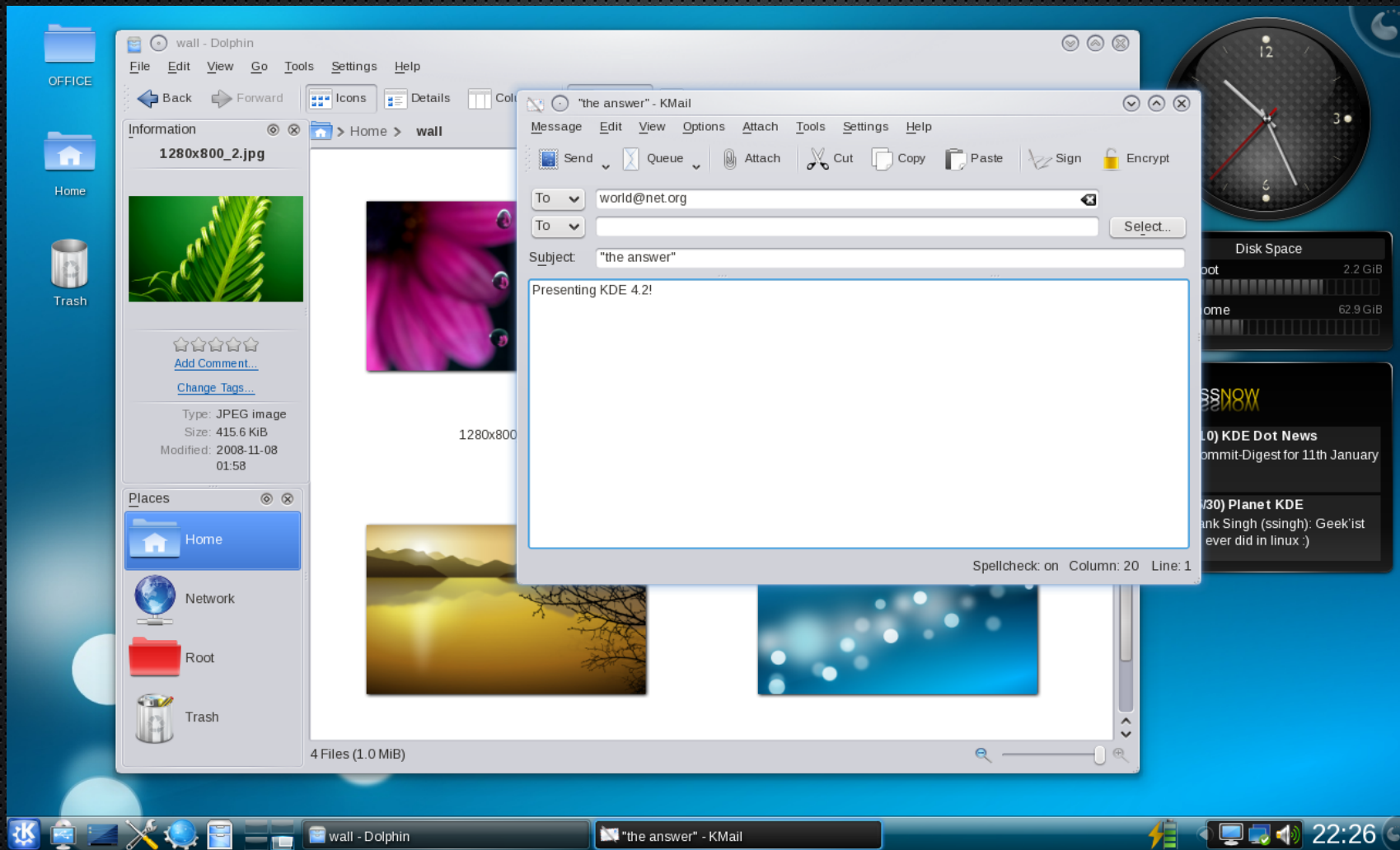
Choices choices choices (for better or worse)

Linux has many different

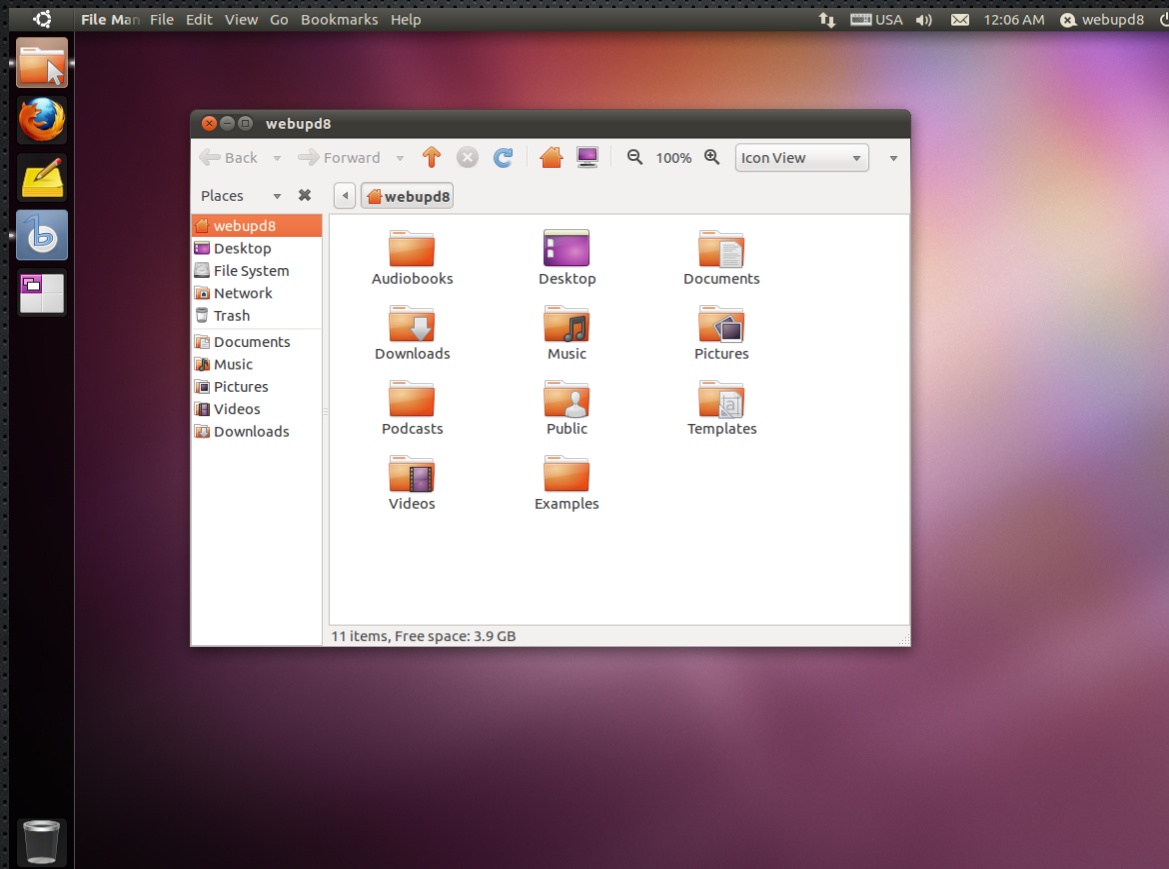
“Desktop Environments”
(or Window Managers)

(which, to most, probably look like completely different operating systems)

KDE



Unity (Ubuntu)

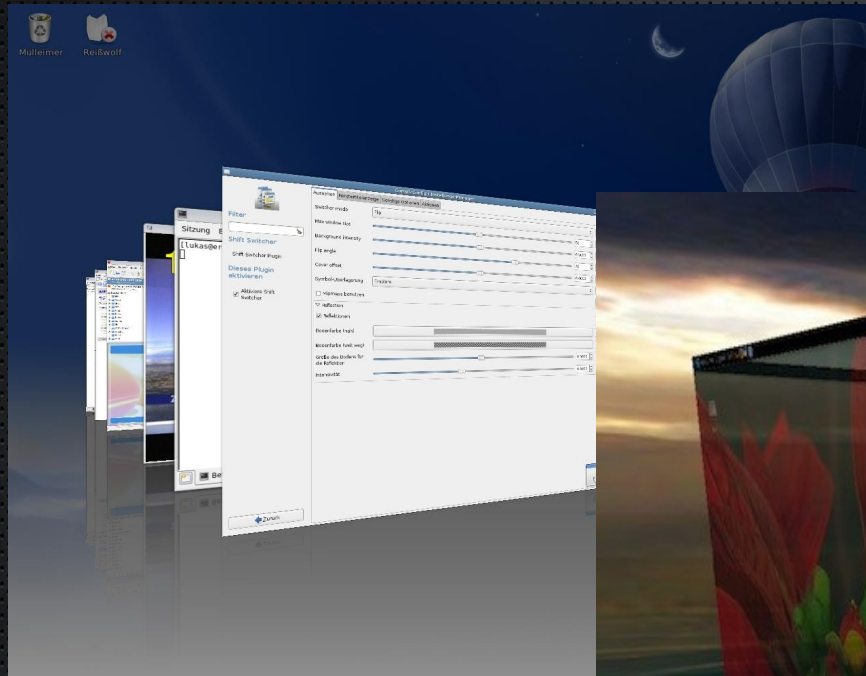


A screenshot of a Linux desktop environment. The background is a scenic image of a lake and trees at sunset. In the foreground, there is a file manager window titled 'Home' showing a sidebar with 'Personal' and 'Devices' sections. The main area displays folders for Documents, Downloads, Music, Pictures, Public, Templates, and Videos. Overlaid on the file manager is a terminal window with the following content:

```
dhani@dhani-freya: /usr/share/icons
+ ...ni-freya: ~/Downloads | X ...freya: /usr/share/icons
dhani@dhani-freya: ~/Downloads$ cd /usr/share/icons
dhani@dhani-freya: /usr/share/icons$ ls
default      handhelds    ubuntu-mono-dark
mono-light   on-theme     snaps-applications
ss
Lion-2 /usr/share/themes
```

At the bottom of the screen is a dock with various application icons including a web browser, email client, messaging app, calendar, music player, system settings, and other utilities.

Fancy compiz fanciness



OpenElec (XBMC/Kodi)



Awesome

about - awesome window manager - Vimperator

index.mdnwn [~/Work/src/awesome/www] - VIM

Mutt with 516 messages

1215630123 time

about - awesome window ma...

awesome

home concepts news download community wiki

A window manager is probably one of the most used software in your day-to-day tasks, with your Web browser, mail reader and text editor. Power users and programmers have a big range of choice between several tools for these day-to-day tasks. Some are heavily extensible and configurable.

awesome tries to complete these tools with what we miss: an extensible, highly configurable window manager.

To achieve this goal, **awesome** has been designed as a framework window manager. It's extremely fast, small, dynamic and heavily extensible using the [Lua](#) programming language.

We provide an easily usable and very-well documented API to configure and define the behaviour of your window manager.

Did you ever imagine press one key and see all your windows arranged automatically?

Did you ever imagine type a window's name and get it back in front of you?

Did you ever imagine *ssh* to a computer and see the load average of this one to be graphed directly in the titlebar of your terminal emulator?

awesome allow you to do that, and more, as long as you can express it in a programming language.

Features and non-features

- × Very stable;
- × Complete and very well documented source code and API;
- × No mouse needed: everything can be performed with keyboard;
- × Real multithread support (XRandR, Xinerama or Zaphod mode);
- × Implement many [Freedesktop](#) standards: [EWMH](#), [XDG Base Directory](#), [XEmbed](#), [System Tray](#);
- × Doesn't distinguish between layers: there is no floating or tiled layer;
- × Whether or not the clients of currently selected tag(s) are in tiled layout, you can rearrange them on the fly. Popup and fixed-size windows are always floating, however;
- × Layout handling: can automatically manage your windows placement according to the chosen policy for each tag;
- × Use tags instead of workspaces: allow to place clients on several tags, and display several tags at the same time;
- × [D-Bus](#) support;
- × And more.

This gonna be LEGEN... wait for it... DARY!

34 Developers

Copyleft 2007-2008, awesome project

Last edited Wed Jul 9 19:44:40 2008

18 We provide an easily usable and very-well documented API to configure and define the behaviour of your window manager.

20

21 Did you ever imagine press one key and see all your windows arranged automatically?

22

23

24 Did you ever imagine type a window's name and get it back in front of you?

25

26 Did you ever imagine *ssh* to a computer and see the load average of this one to be graphed directly in the titlebar of your terminal emulator?

27

28

29 **awesome** allow you to do that, and more, as long as you can express it in a programming language.

30

31

32 **Features and non-features**

33 **×** Very stable, fast, small and simple;

34 **×** Complete and very well documented source code and API;

35 **×** No mouse needed: everything can be performed with keyboard;

36 **×** Real multithread support (XRandR, Xinerama or Zaphod mode);

37 **×** Implement many [Freedesktop](#) standards:

38 [EWMH](#) ([http://standards.freedesktop.org/ewmh-spec/ewmh-spec-latest.html](#)),

39 [XDG Base Directory](#) ([http://standards.freedesktop.org/basedir-spec/basedir-spec-latest.html](#)),

40 [XEmbed](#) ([http://standards.freedesktop.org/xembed-spec/xembed-spec-latest.html](#))

41 [System Tray](#) ([http://standards.freedesktop.org/systemtray-spec/systemtray-spec-latest.html](#));

42 **×** Some real transparency support (using Composite extension and xcompmgr);

43 **×** Doesn't distinguish between layers: there is no floating or tiled layer;

44 **×** Whether or not the clients of currently selected tag(s) are in tiled layout,

45 you can rearrange them on the fly. Popup and fixed-size windows are

46 automatically floating.

47 **×** Layout handling: automatically manage your windows placement according to

48 the chosen policy for each tag;

49 **×** Use tags instead of workspaces: allow to place clients on several tags, and

50 display several tags at the same time;

index.mdnwn [ikiwiki] 33, 37 802

491 SF Jul 01 To awesome@naqu (0.8K) ↳Re: [PATCH] Add a tags target, cmake alrea

492 r Jul 02 Marco Candrian (1.6K) [PATCH] [widgets/progressbar] add fg_off to s

493 S Jul 02 Marco Candrian (1.4K) ↳Re: [PATCH] [widgets/progressbar] add fg_o

494 rs Jul 02 Marco Candrian (1.4K) ↳

495 SF Jul 02 To awesome@naqu (1.6K) ↳

--Mutt: slist-awesome/ [Msgs:516 19M]---(threads/date)-----(952)---

From: calmar <mac@calmar.us>

To: awesome@naquadah.org

Subject: Re: [PATCH] [widgets/progressbar] add fg_off to set + fix copy/paste typo

Date: Wed, 2 Jul 2008 02:54:42 +0200

Mail-Followup-To: calmar <mac@calmar.us>, awesome@naquadah.org

User-Agent: Mutt/1.5.17 (2008-04-09)

-- PGP output follows (current time: Wed Jul 9 21:01:38 2008) --

gpg: Signature made Wed Jul 2 02:54:42 2008 CEST using DSA key ID 53D90F4D

gpg: Good signature from "Marco Candrian <mac@calmar.us>"

gpg: WARNING: This key is not certified with a trusted signature!

gpg: There is no indication that the signature belongs to the owner.

Primary key fingerprint: 5514 F43C B0CC 584C 556A B8F2 F6B4 834C 53D9 0F4D

-- End of PGP output --

-- The following data is signed --

On Wed, Jul 02, 2008 at 02:48:35PM +0200, marco candrian wrote:

Hi all,

there is another issue with the progressbar (probably also with

the graph).

area_t is used as something like a vector - for the color-gradient,

Therefore width and height can have negativ values, and therefore

should not but unsigned, what area_t is so,

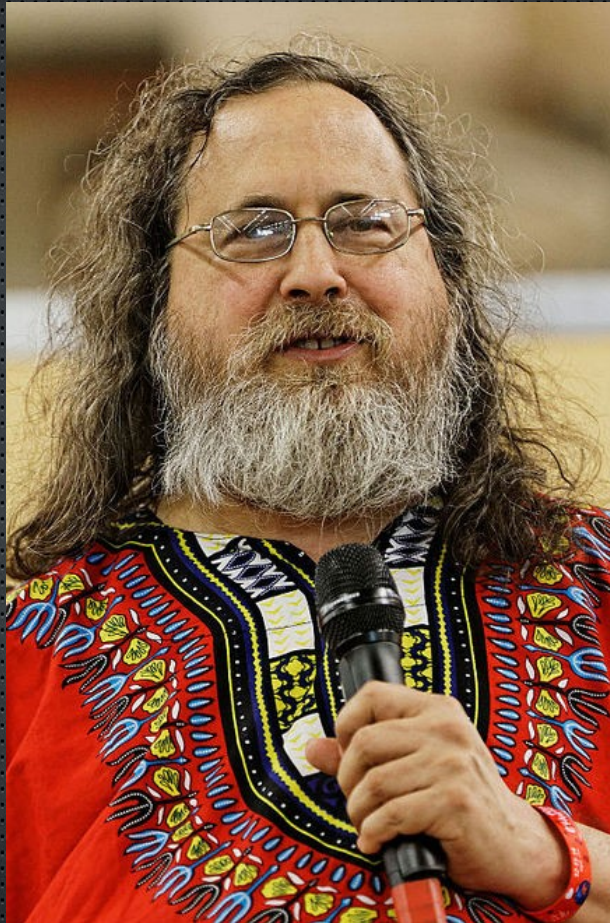
So, either something different to area_t as a 'vector' has to be

- S - 493/516; Marco Candrian Re: [PATCH] [widgets/progressbar] add -- (69%)

PGP signature successfully verified.

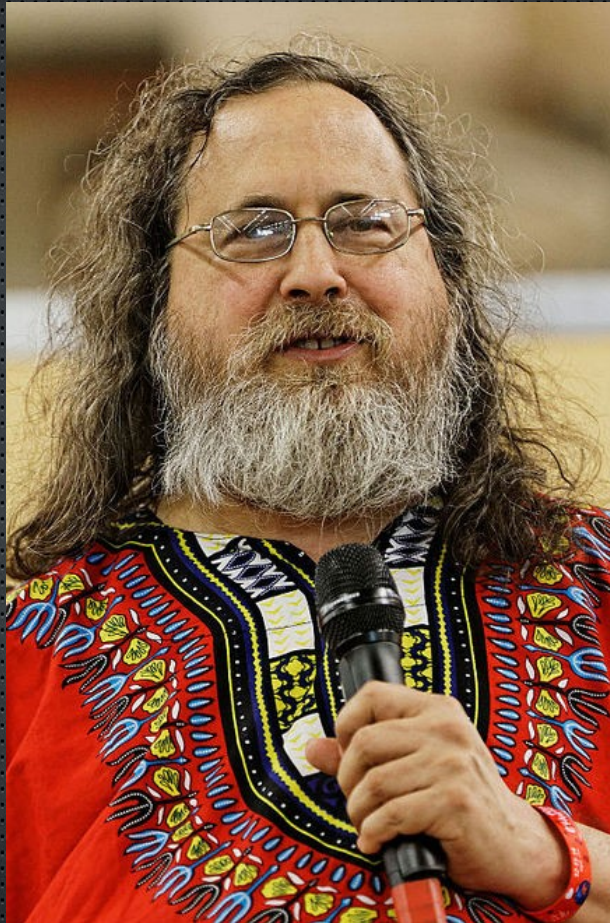
Richard Stallman:

Weird, picky guy who says crazy things



Richard Stallman:

Weird, picky guy who says crazy things



THAT ALWAYS END UP BEING TRUE IN THE LONG RUN

Richard Stallman:

Weird, picky guy who says crazy things,
e.g

They're going to go into your home
and burn your books!!

Richard Stallman:

Weird, picky guy who says crazy things,
e.g

They're going to go into your
COMPUTER
and burn your FILES!!

That's impossible, unless?

they kill the file manager

Hello iTunes! Hello Amazon!

Why Command Line?

Because you can very quickly say/relate complex concepts in a concise way, by combining a series of simple symbols.

You know, like talking. Or writing.

Command line is the act of literally talking to the computer....unlike...

What's so bad about the mouse again?

Not **bad**, just simple: “Caveman interface,” you can only point and grunt.

(Tablets and even “Minority Report” are cool and fun...but why is Charades a game?)

Various names for the stuff we do today:

Command line: Blinky cursor area that's literally asking you, “okay, now what?”

Terminal: App for command line (used to be the computer itself)

Shell: Any particular “type” of command-line environment. Examples are Bash, Fish, Zsh, MS-Dos, etc.

Bash: “Bourne Again Shell; the specific Linux/Unix shell we will use.

Scripting: Putting a bunch of shell commands in a file and running it as a program.

Why isn't “scripting” a bigger thing?

Why is “scripting” NOT considered “real programming?”

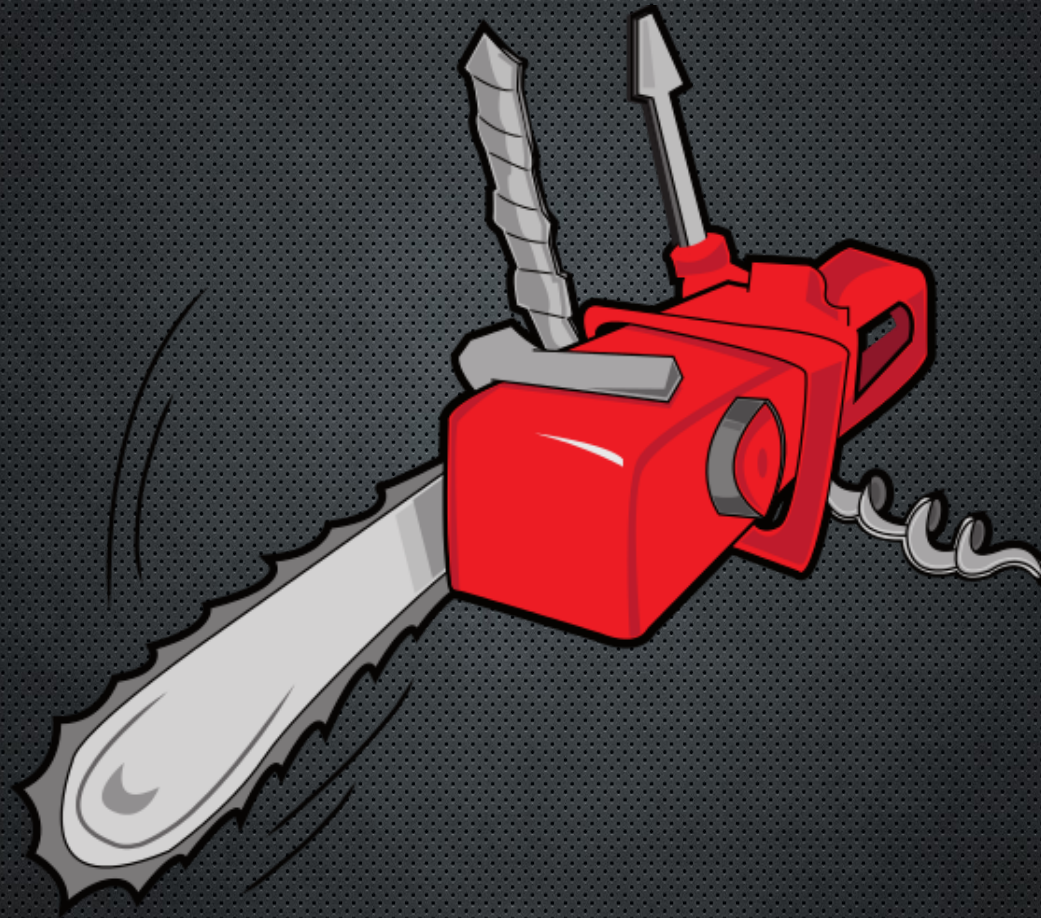
- Quick and dirty
- Performance can be slow (especially as compared to compiled)
- Lacks “libraries” or “frameworks”
- Few tools/structures designed for reuse or collaboration

“Real” Programming Languages

(C, Python, Ruby, Java etc.)



Bash/Shell scripting



For example...

```
sudo rm -rf /
```

- seriously, don't do this.

Users and Permissions

(they actually mean something here)

ROOT – Like “Administrator” or maybe “God”

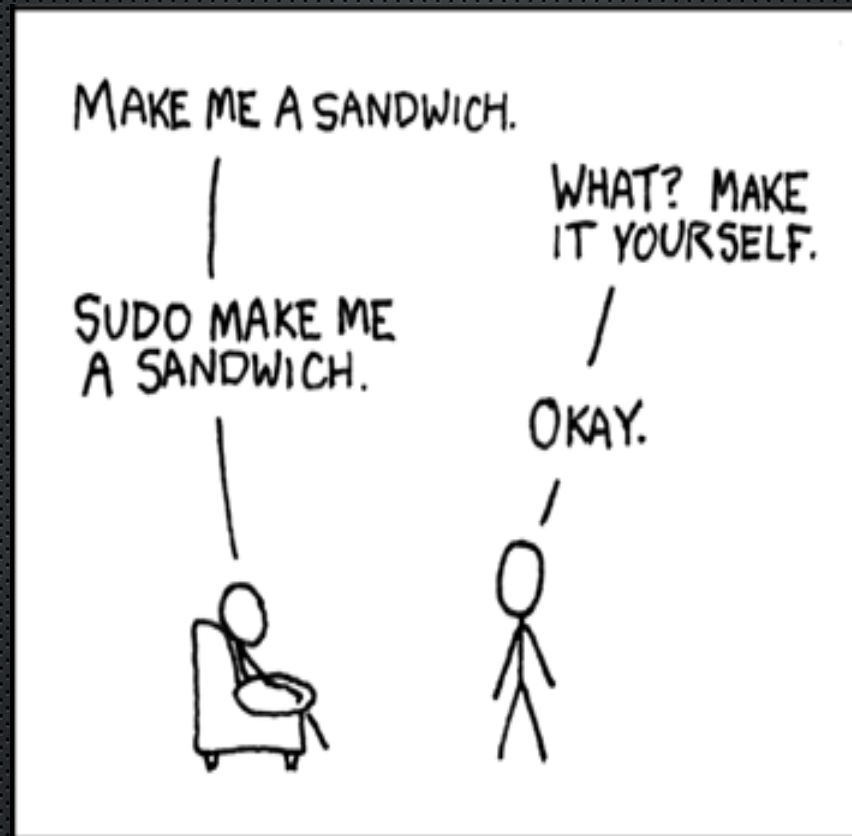
users – humans

(..and others – fake “users” to get tasks done)

Some systems (eg Ubuntu) allow for Super Users

S.U.- do “this” = `sudo`

And now...this makes sense



Why Linux has no virus problem

Windows historically does not distinguish between:

files you're meant to read/watch/hear/edit, and
files you're meant to **run**.

A piece of paper that says “Go jump off a bridge” is pretty harmless...unless....

P.S.

In this set of slides, I will **not** test you on anything from here, forward...

Permissions

aka why original windows was amazingly stupid because multiple people might want to sometimes use the same computer

Three major things you can do with files

READ (look at, view, listen to)

WRITE (and delete and edit)

EXECUTE (run as a program)

Three important “groups”

owner of the file

owner's group

everybody else

Permissions

Quick note on permissions for directories (kind of non-intuitive)

READ: Is able to read the directory listing

WRITE: Is able to change contents of the directory

(create new/delete existing files, or rename them)

EXECUTE: Is able to access/ go to the directory

Permissions

(that funky line when you do a `ls -l`)

0123456789

-rwxr--r--

dr-x-----

Permissions

(also, how computers work)

- | Octal | Text | Binary | Meaning |
|-------|------|--------|--------------------------------------|
| 0 | --- | 000 | All types of access are denied |
| 1 | --x | 001 | Execute access is allowed only |
| 2 | -w- | 010 | Write access is allowed only |
| 3 | -wx | 011 | Write and execute access are allowed |
| 4 | r-- | 100 | Read access is allowed only |
| 5 | r-x | 101 | Read and execute access are allowed |
| 6 | rw- | 110 | Read and write access are allowed |
| 7 | rwX | 111 | Everything is allowed |

Permissions

Thus – permission types like

644

oge

owner can read and write (4+2)

group can only read (4)

others can only read (4)

Practical Permission problems you are likely to encounter:

- If you're unable to view, execute, or delete/change a file, try this.
- If you write a little shell script (.sh), remember to set it executable. (The only permission command I use on a regular basis is `chmod +x "file.sh"`)
- FAT and NTFS filesystems (the ones Windows use) don't have permissions, but Linux has to occasionally pretend they do, this causes problems.
- When you're taking a website online, this is often a difficult issue. (For a good reason; you don't want website visitors overwriting your critical files!)

File Paths

File paths are HIERARCHICAL and DELIMITED by backslashes, starting with root, at “/”, e.g.

```
/media/cdrom/mypaper.txt
```

signifies a file “mypaper.txt” in a folder called “cdrom”, and THAT folder is in a folder called media – and “media” is in the root directory.

SPECIAL FOLDERS:

~ or ~/ signifies the user's home folder. i.e. if your username is fsmith, and you are logged in: ~/ = /home/fsmith/

. (one period) refers to your current folder

..(two periods) refers to one folder up. Thus, if you're currently in /home/fsmith then ../ would refer to /home.

The LINUX Filesystem

(EVERYTHING is a file!)

/bin, /sbin - Systemwide binaries

/boot - Boot Stuff

/dev - devices

/etc - (Some) helper files

/home/user - YOUR files & config (you can just back this up)
 .files (dotfiles)

/lib - Libraries (kind of like dlls)

/lost+found - improper shutdown?

/opt - non-default/weird programs

/mnt, /media - generic "mount points"

/proc - the actual running processes whooa

/usr - User stuff (mostly binaries)

/tmp - temp files

/var - other spooling data, logs

Getting help

`man` (command)

`info` (might give you more info)

`apropos` (keyword to search)

`help` (pretty basic stuff)

but seriously, Google/Duckduckgo etc

Linux/Unix Commands (verbs)

Any action or program the computer can do

Commands often have ARGUMENTS, either:

OPTIONS (adverbs)

- One dash + a letter (ls -a)
- Two dashes + words (sort --reverse)

EXPRESSIONS (nouns)

- Text, numbers, files, streams, anything you want to manipulate

File Manipulation

ls - list

cd - change directory

rm -remove (delete for good)

mv - move OR rename (they are literally the same thing, weird)

cp - copy

Viewing text and files

`cat` - “concatenate”

`less` - this is such a terribly bad joke I hate even explaining it

...but what about editing?

Editing Files

`nano/pico` (text-based, “normal” keys)

`vi/vim` (hardcore choice 1 universal,modal)

`emacs` (hardcore choice 2)

When you turn on your computer

- 1) Electricity and Magic
- 2) BIOS/EFI/**UEFI**
- 3) Bootloader (Grub or windows)
- 4) Operating System

Multiple commands, one line

& - Run both simultaneously

&& - Run the first one, and then the second
ONLY IF the first “succeeds,” otherwise
stop.

; - Run the first one, then the second
regardless of what happens.

Even MORE command line.

One quick command I totally forgot:

`echo`

(puts argument through stdout)

Pipes and redirects

Default behavior:

read from “stdin”, write to “stdout”

- > (over)write/replace a file
- >> write to/append to file
- < read from file
- | pipe output from first command into 2nd
- tee pipe AND write to stdout

BASH

BASH (Bourne Again) Shell - others are fish and zsh, etc

Lots of “tricks” are available here, eg

- Tab completion
- Up arrow key for history
- Ctrl-R to search history

and many MANY more

More BASH

Furthermore, you can modify this environment to fit your needs, via:

`.bashrc`

(stuff here will be run everytime you open a terminal)

A great example is the “alias” command. If a command doesn't exist for what you want to do, just make up your own!

```
alias modbash='gedit ~/.bashrc'
```


Linux/Unix Commands

An action or program that a computer can do

Find them with “apropos,” learn about them with “man”

(check these out <http://www.oreillynet.com/linux/cmd/>)

Commands can optionally have ARGUMENTS, in the form of:

OPTIONS

one dash + letter (`ls -a`)

two dashes + words (`sort --reverse`)

EXPRESSIONS

text; numbers; files; streams – things to be manipulated

Opening Files

IN TERMINAL

`less`

`cat (stdout)`

COMMAND/ARGUMENT STYLE

`gnome-open file`

`vim textfile`

`firefox localfile.html`

`firefox http://slashdot.org`

SORT

- - i = case INSENSITIVE
- - r = REVERSE
- - g = numbers
- - R = random

GREP (line matching)

```
grep OPTIONS PATTERN (FILE)
```

Can search over FILES or STDIN

Also, can search ONE FILE or MANY (check -d or -R)

useful flags:

-i (case insensitive)

-v (invert search/show NON-matches)

-l (just show matching FILES, not lines)

FIND (files)

Searches directory tree rooted at given filename (default current)

Good if you also want to use parameters like “date”, “last accessed”, “size” and so forth.

Often used with -name or -iname

Also, consider “locate” (database must be setup beforehand)

SED (stream editor)

Considered an entire language

Usually used with “s” for substitution

Delimiters are usually slashes but can be anything

REGULAR EXPRESSIONS

```
echo “Good day” | sed 's/day/night/'
```

<http://www.grymoire.com/Unix/Sed.html>

<http://sed.sourceforge.net/sed1line.txt>

AWK

```
awk <search pattern> {<program actions>}
```

Also a text-processor, good for flat-file databases

Also, an entire language

```
awk ' /apples/ { print $2 " " $1 } '
```

<http://www.vectorsite.net/tsawk.html>

<http://www.pement.org/awk/awk1line.txt>

CLI v GUI?

- Command Line Interface
-
- Vs
-
- Graphical User Interface

.....why not both?

CLI, but GUI-ish

- Nano
- Mc (midnight commander)

From CLI to GUI

- Opening file on command line

`firefox home.html`

(Remember, closing the terminal will also close the program)