

Session 4: tmux



P. S. Langeslag

8 November 2018

Vim on Your Own System

OS X

Included; open a terminal and enter `vim`

Linux

Included or in package repositories; install the `gvim` package for full clipboard functionality

Windows

Download from <http://www.vim.org>

Terminal or Graphical?

Whatever works for you

Display Protocols

1984–	X Window System	Still the default display protocol for UNIX-like systems
2008–	Wayland	Abandons 1980s standards in pursuit of greater efficiency and security

Graphical Environments

Desktop Environment (DE)

Complete graphical environment, with windows, icons, sound effects, themes, and graphical interfaces to various tools. (Gnome, KDE, Cinnamon, MATE, Xfce)

Window Manager (WM)

Does what it says and little else. All other DE components can be separately installed.

Display Manager

Graphical application for logging a user into Linux and launching the desktop environment or window manager.

Display Manager: Gnome Display Manager (GDM)

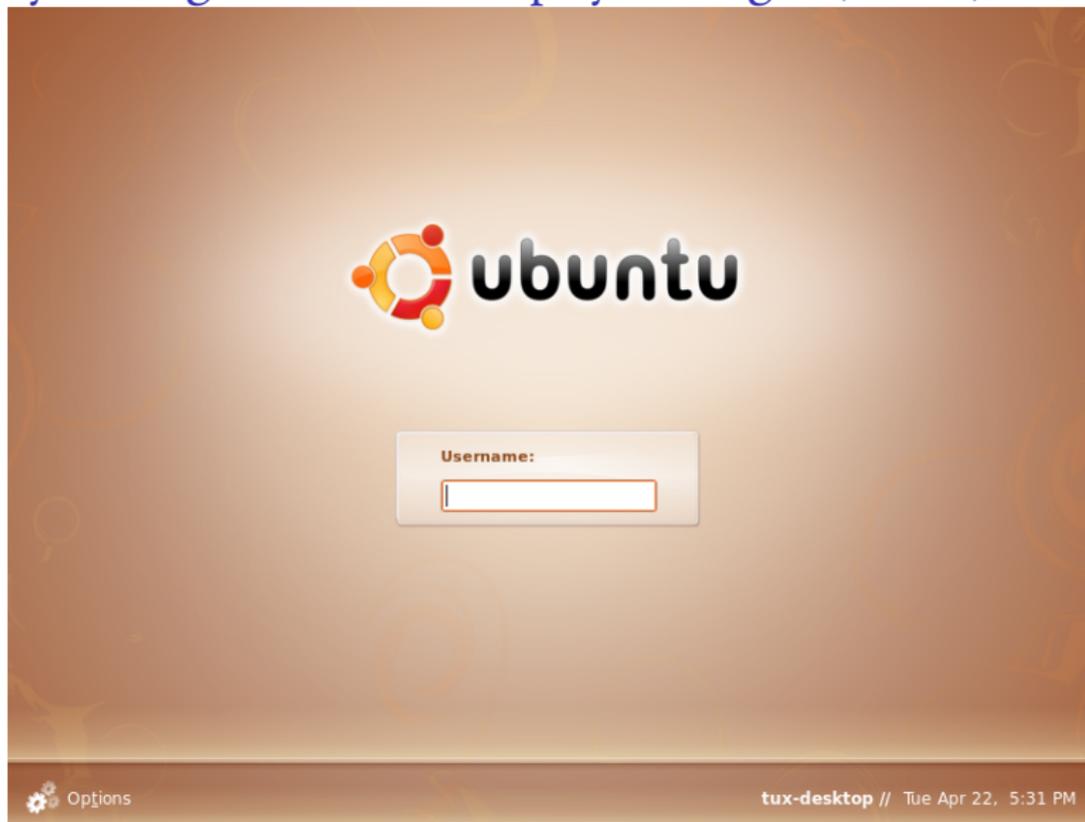


Figure 1: GDM on Ubuntu 10.04 (public domain / [WMC](#))

Desktop Environments: Gnome



Figure 2: Gnome 3 (<https://www.gnome.org>)

Desktop Environments: KDE



Figure 3: KDE Plasma (<https://www.kde.org>)

Desktop Environments: MATE

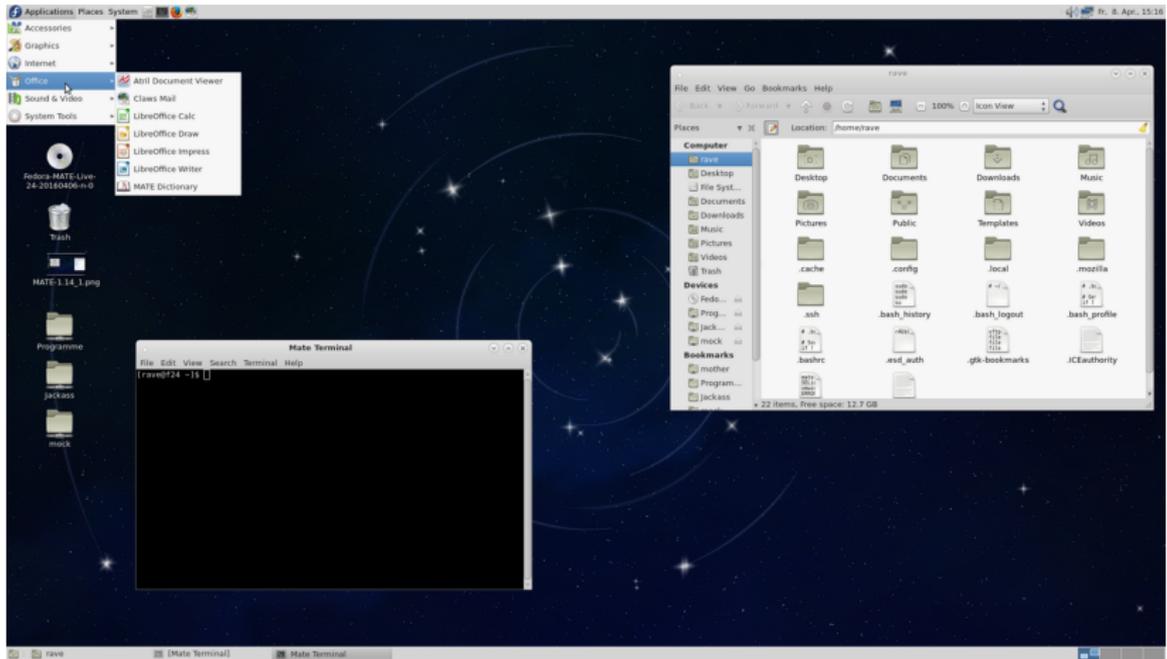


Figure 4: MATE 1.14 (<https://www.mate-desktop.org/gallery/>)

Desktop Environments: Xfce

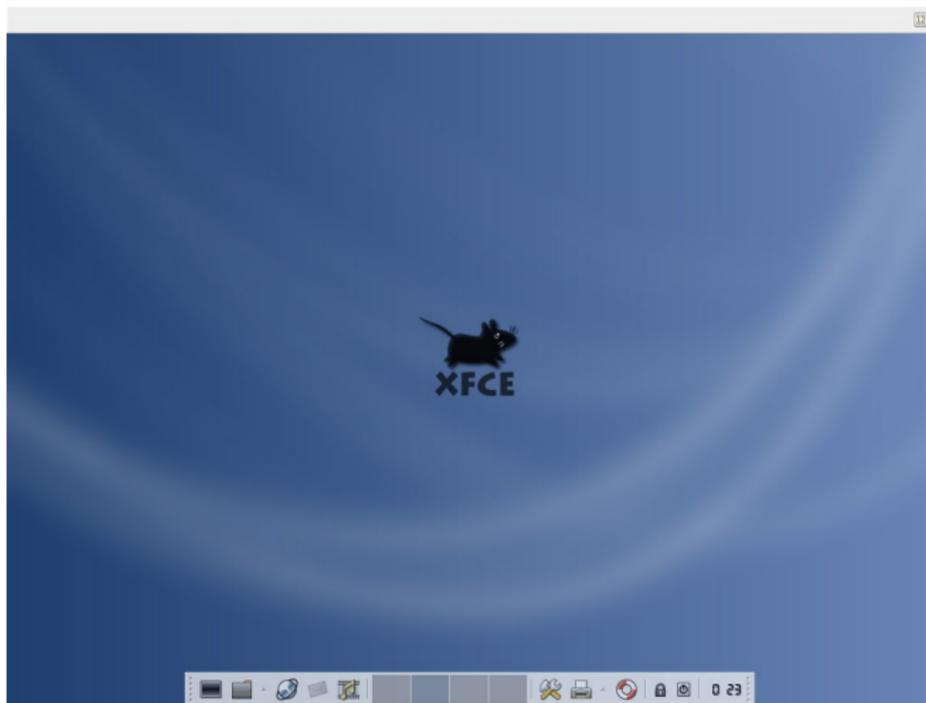


Figure 5: Xfce 4.0 (<https://www.xfce.org/about/screenshots/>)

Window Management

Stacking (floating)

Allows windows to overlap and be positioned freely, each on its own layer.

Tiling

Positions windows automatically on a single layer following a fixed algorithm.

Dynamic

Provides multiple, switchable tiling layouts.

(Dynamic Tiling) Window Managers: i3

The screenshot displays the i3 window manager interface. On the left, a terminal window shows the source code for `handlers.c`, which includes a `handle_xap_request` function. The code handles XAP requests by checking for cookies and logging errors. The terminal output shows the compilation of `handlers.o` and the execution of `nmcli` to set network settings. Below the terminal, a video player window is open, showing a scene from the movie *The Big Bang Theory* featuring Sheldon Cooper.

On the right, a browser window displays the i3 homepage. The page title is "i3 - an improved dynamic tiling window manager". It features a navigation menu with links for [Goals](#), [Documentation](#), [Downloads](#), and [Contact](#), along with a link to [Impressum/Imp rint](#). The main text explains that i3 was created because `wmii` didn't provide some features and had bugs. It lists four goals for i3: 1. Write well readable, well documented code. 2. Use xcb as far as possible. 3. Implement Xinerama correctly. 4. Use the metaphor of a table for abstraction. The footer of the browser window shows the URL `http://i3.zekjur.net/` and a "Top" button.

Figure 6: i3 (<https://i3wm.org/#/screenshots/>)

(Dynamic) Tiling Window Managers: awesome

The image shows a terminal window with two panes. The left pane displays a git log with columns for date, author, commit hash, and message. The right pane shows the content of a .zsh-theme file, which includes color definitions, cursor styles, and various prompts for different shell states.

```
2017-01-12 23:03 Daniel Hahler o [vim-8.0.0121] [bluayed/vim-8.0.0121] Linting
2017-01-12 22:20 Daniel Hahler o vim-patch:8.0.0121
2017-01-13 04:09 Justin M. Keyes M [master] [origin/HEAD] [origin/master] Merge #5933 fr
o Windows: enable more tests
o toki: BufEnter
o lint:
o open_buffer(): Do 'BufEnter' for directories.
2017-01-12 18:24 James McCoy M Merge pull request #5935 from jamessan/dictwatcher-
o eval: Remove dictwatcher from watchers, come before
[patch-8.0.0121-cursorline-pos] Merge pull request
2017-01-12 06:45 James McCoy M o Inconmann: Preview :isub commands only after the dell
o Inconmann: Suppress error reporting when previewing
2017-01-11 06:00 Justin M. Keyes M o Windows: vim.getenv(): Find runtime relative to nvim.
2017-01-11 02:19 Justin M. Keyes M Merge #5910 from justinmk/win32-jobstart
o test: system(...): v:shell_error
o system(...): Set v:shell_error=1 if not executable
2016-09-30 15:29 Rui Abreu Ferreira o Windows: ci/Applyenv: Enable Python provider tests
2016-08-25 17:28 Rui Abreu Ferreira o test: system(...):
2016-08-27 01:27 Rui Abreu Ferreira o Windows: libuv_process_spawn(): Allow libuv argument
2017-01-11 00:50 Marco Hinz o clipboard: only check for gpbcopy on macOS (#5927)
2017-01-10 15:32 James McCoy M Merge pull request #5924 from jamessan/vim-7.4.2100
o lint
o vim-patch:7.4.2100
2017-01-10 10:21 James McCoy M Merge pull request #5863 from Zyk-I/nore-clint-checks
o clint: Check for misplaced brace at function start
o clint: Enable check for | positioned at the start o
2017-01-10 09:56 James McCoy M Merge pull request #5919 from jamessan/vim-7.4.2088
o lint
2017-01-09 13:19 James McCoy o vim-patch:7.4.2009
2017-01-08 23:13 James McCoy o vim-patch:7.4.2008
2017-01-09 22:31 James McCoy M Merge pull request #5862 from jamessan/vim-aab315d
o vim-patch:c95a302
2017-01-07 14:09 James McCoy o vim-patch:7.4.1975
2017-01-01 23:25 James McCoy o vim-patch:82a7f71
2017-01-01 20:48 James McCoy o vim-patch:aab315d
2017-01-09 20:20 James McCoy M Merge pull request #5860 from jamessan/vim-7.4.1889
o vim-patch:7.4.1889
2017-01-09 07:22 Justin M. Keyes o process_wait(): Avoid dereference after LOOP_PROCES
2017-01-09 00:07 Justin M. Keyes M Merge #5919 from bluayed/improve-python-health-ch
o [improve-python-health-check] [bluayed/improve-py
o healthcheck: s:check_python: only report latest w
o healthcheck: python: prefer neovim, VERSION
o healthcheck: python: include nvim path for unknow
o Improve error reporting for Python health check
2017-01-09 16:46 Daniel Hahler o s:check_python: handle pip install --user -e
o health: completion for :checkhealth
2017-01-08 16:23 Daniel Hahler o health: Windows: Handle backslash path separators
2017-01-08 21:26 Justin M. Keyes M Merge #5909 from justinmk/win32-xdg
o test: Windows: Remove shade functional tests
2016-09-31 13:19 Rui Abreu Ferreira o [main] #588165485120c08665afca2a0323bc69f19 commit of S
```

```
{ -o/nvim }
^ vim-8.0.0121@x3 cd awesome
~/src/awesome
- ~/src/awesome -
^ build-utils-travis-a-@x3 } gfa
Fetching origin
remote: Counting objects: 15, done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 15 (delta 3), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (15/15), done.
From github.com:awesomeM/awesome
60769f7f1bc085d2 master -> origin/master
Fetching psychon
Fetching Elv13
remote: Counting objects: 227, done.
remote: Compressing objects: 100% (171/171), done.
remote: Total 227 (delta 110), reused 8 (delta 8), pack-reused 38
Receiving objects: 100% (227/227), 55.89 KiB | 0 bytes/s, done.
Resolving deltas: 100% (131/131), completed with 5 local objects.
From git://github.com/Elv13/awesome.1
* d0b9195b...c08e1cbb upstream_dynamic_p7 -> Elv13/upstream_dynamic_p7 (forced update)
^ build-utils-travis-a-@x3 }
(c) (M) %?
```

```
1|--oniz2/bluayed-zsh-theme [~/neovim]
color=${cwidth}
colored=${(color)${(fg/BN)}/${(color.off)}
cursor='
done
cwidth=${(j):/}colored
# Display repo and shortened revision as vcs info, if available.
# Add exclamation mark for "fresh"
if [[ -n $vcs_info_msg_1 ]]; then
local indicator
[[ -n $ZSH_VCS_INFO_FORCE_GETDATA ]] && indicator='!'
rprompt_extra+=("${(repeat)${(vcs_info_msg_1)_indicator}")}
# TODO: if cwidth is too long for COLUMNS-restofprompt, cut longest parts of cwd
rprompt_cwd=${(htxt)100$ZSH...-${(cwidth)k}k}
prompt_cwd=${(PR_RESET)k} ${(cwidth) $({PR_RESET})}
# user@host for SSH connections or when inside an OpenVZ container.
local remote
~/dotfiles/oh-my-zsh/themes/bluayed-zsh-theme 1587L, 57923C written
```

Figure 7: awesome <https://github.com/awesomeWM/awesome/issues/1395>

(Dynamic Tiling) Window Managers: dwm

```
1 2 3 4 5 6 7 8 9 []= anselm@garber: ~
38 static unsigned int doubleclicktimeout = 300;
39 static unsigned int tripleclicktimeout = 600;
40
41 /* alt screens */
42 int allowaltscreen = 1;
43
44 /* frames per second st should at maximum draw to the screen */
45 static unsigned int xfps = 120;
46 static unsigned int actionfps = 30;
47
48 /*
49  * blinking timeout (set to 0 to disable blinking) for the terminal
    linking
50  * attribute.
51  */
52 static unsigned int blinktimeout = 800;
53
54 /*
55  * thickness of underline and bar cursors
56  */
57 static unsigned int cursorthickness = 2;
58
59 /*
60  * bell volume. It must be a value between -100 and 100. Use 0 for d
    sabling
61  * it
62  */
63 static int bellvolume = 0;
64
65 /* default TERM value */
66 char *terminame = "st-256color";
67
68 /*
69  * spaces per tab
70  */
71 * When you are changing this value, don't forget to adapt the *it=
    value in
72 * the st.info and appropriately install the st.info in the environme
    nt where
    38,1 8%
```

goals/	openbsd
patches/	Description
alpha/	
anysize/	
clipboard/	OpenBSD primarily searches for terminfo descriptions in terminfo databases before considering terminfo files. Given the terminfo currently stored in the global database is for st 0.1.1, this leads to conflicts and misbehaviour.
copyurl/	
delkey/	
disable bold italic fonts/	
dracula/	This patch renames st to st-git forcing OpenBSD to use the provided terminfo file.
externalpipe/	
fix keyboard input/	Notes
hidecursor/	Once a new stable version of st is out, the corresponding changes to st.info can be pushed
iso14755/	
keyboard	

```
IG -DUSER_PCICONF -DAPERTURE -DMTRR -DNTFS -DHIBERNATE -DPCIVERBOSE -DUSB
VERBOSE -DWSDISPLAY_COMPAT_USL -DWSDISPLAY_COMPAT_RANKBD -DWSDISPLAY_DEFA
ULTSCREENS=6* -DX86EMU -DONEWIREVERBOSE -DMULTIPROCESSOR -DMAUSERS=80 -
D_KERNEL -MD -MP -c /usr/src/sys/netinet/inet_nat64.c
cc -g -Werror -Wall -Wimplicit-function-declaration -Wno-uninitialized -
Wno-pointer-sign -Wframe-larger-than=2047 -Wno-address-of-packed-member
-Wno-constant-conversion -mmodel=kernel -mno-red-zone -mno-sse2 -mno-sse
-mno-3dnow -mno-mmx -msoft-float -fno-omit-frame-pointer -ffreestanding
-fno-pie -mretpoline -O2 -pipe -nostdinc -I/usr/src/sys -I/usr/src/sys/a
rch/amd64/compile/XP513.MP/obj -I/usr/src/sys/arch -DDDB -DDIAGNOSTIC -DK
TRACE -DACCOUNTING -DKMEMSTATS -DPRTRAC -DPOOL_DEBUG -DCRYPTO -DSYSVMSG
-DSYSVSEM -DSYSVSHM -DUVM_SWAP_ENCRYPT -DFFS -DFFS2 -DFFS_SOFTUPDATES -DUF
S_DIRHASH -DQUOTA -DEXT2FS -DMFS -DNFSCLIENT -DNFSSERVER -DCD9660 -DUDF
-DMSDOSFS -DFIFO -DFUSE -DSOCKET_SPLICE -DTCP_ECN -DTCP_SIGNATURE -DINET6
-DIPSEC -DPPP_BSDCOMP -DPPP_DEFLATE -DPIEX -DMROUTING -DMPLS -DBOOT_CONF
IG -DUSER_PCICONF -DAPERTURE -DMTRR -DNTFS -DHIBERNATE -DPCIVERBOSE -DUSB
VERBOSE -DWSDISPLAY_COMPAT_USL -DWSDISPLAY_COMPAT_RANKBD -DWSDISPLAY_DEFA
ULTSCREENS=6* -DX86EMU -DONEWIREVERBOSE -DMULTIPROCESSOR -DMAUSERS=80 -
D_KERNEL -MD -MP -c /usr/src/sys/netinet/inet_ntop.c
```

Figure 9: dwm (<http://dwm.suckless.org/screenshots/>)

tmux: Terminal Multiplexer

- ▶ Allows local and remote users to detach, reattach their sessions
- ▶ Automatically stores a session when the connection is lost
- ▶ Can be used as a dynamic window manager in the terminal

tmux Pane Navigation

Any tmux commands are preceded by CTRL+B.

"	Split pane into two, top and bottom
%	Split pane into two, left and right
o	Cycle between panes (or use cursor keys)
CTRL+0	Rotate panes
CTRL+cursor	Resize panes slightly
ALT+cursor	Resize panes more drastically
z	Toggle zoom to exclusive view
ALT+1~5	Layouts
SPACE	Cycle layouts
x	Kill pane (or type <code>exit</code> in shell)

tmux Window Navigation

c	Create new window
0~9	Move to window 0-9
n	Cycle between windows
&	Kill window
:kill-window -a	Kill all windows except the active one
, (comma)	Rename window

tmux Session Commands

\$	Rename session
d	Detach client
& on last window	Kill tmux session (or enter <code>exit</code> in shell)

In the shell, `tmux kill-server` kills all sessions and closes tmux.

tmux Reattach and Session Name Options

<code>tmux attach</code>	Reattach the most recent lost or detached session
<code>tmux new -s name</code>	Start a new session called name
<code>tmux attach -t name</code>	Reattach a previously detached session called name
<code>tmux ls</code>	List current sessions (attached or detached)

Paste Buffer and Commands

[Enter copy mode
SPACE	Start selection (no CTRL+B prefix)
ENTER	Copy selection (no CTRL+B prefix)
q	Quit mode (no CTRL+B prefix)
]	Paste from buffer
:	Open tmux command prompt

(NB in the default emacs mode, it's CTRL+SPACE, CTRL+W for start and copy selection.)

The command prompt allows such tmux commands as

```
rename-window newname  
new-window -n '.bashrc' vim ~/.bashrc
```

These commands can also be invoked as tmux command-line options.

Configuration

~/tmux.conf, e.g.

```
set -g prefix C-a
```

```
unbind C-b
```

```
bind-key C-a send-prefix
```

```
bind-key C-l clear-history
```

```
set -g status-style bg=colour24
```

```
set -g status-right ""
```

```
set-option -g set-titles on
```

```
set-option -g set-titles-string "#W"
```

bash Scripting

```
#!/bin/bash           # This line identifies the shell interpret-  
                      # er; it should always be your first line.  
  
echo "Hello world"   # The remainder of your file can simply be  
                      # a sequence of things you'd otherwise writ  
                      # on the command line.  
  
                      # Hashes signal comments.
```

Make your file executable as follows:

```
chmod +x filename
```

Then run it with a leading path, as I haven't added any of your directories to \$PATH:

```
./filename
```

A Simple bash Script

```
s3cmd sync ~/admin s3://backups/ --delete-removed -v  
s3cmd sync ~/research s3://backups/ --delete-removed -v  
s3cmd sync ~/teaching s3://backups/ --delete-removed -v  
s3cmd sync ~/html s3://backups/ --delete-removed -v
```

- ▶ Just a sequence of commands that will work in all shells
- ▶ Thus no need to identify the shell interpreter

Variables

```
#!/bin/bash
```

```
string="Hello World!"
```

```
echo $string
```

- ▶ Set a text variable as name="text", no spaces and no \$
- ▶ Call it as \$name, e.g. echo \$name

Command Substitution

- ▶ Allows the output of a command to replace the command itself, so it can be processed more straightforwardly.
- ▶ Enclose the command in `$(...)` and assign it to a variable.

Command Substitution

- ▶ Allows the output of a command to replace the command itself, so it can be processed more straightforwardly.
- ▶ Enclose the command in `$(...)` and assign it to a variable.

```
#!/bin/bash
```

```
state=$(cat /sys/class/power_supply/AC/online)  
echo $state
```

Integer Arithmetic: `let`

- ▶ Handles arithmetic evaluation
- ▶ No spaces unless you use quotes

```
#!/bin/bash
```

```
let addition=2+2  
let subtraction=2-2  
let multiplication="2 * 2"  
x=10  
y=2  
let division=$x/$y  
echo $addition           # etc.
```

Integer Arithmetic: Arithmetic Expansion

- ▶ Handles evaluation and command substitution
- ▶ Spaces permitted

```
#!/bin/bash
```

```
addition=$((2+2))  
subtraction=$(( 2 - 2 ))  
echo $(( 2 * 2 ))  
x=10  
y=2  
division=$(( $x / $y ))  
echo $addition           # etc.
```

A Simple Integer Arithmetic Script

```
#!/bin/bash
```

```
read -p "Enter two numbers separated by a space: " x y  
ans=$(( x + y ))  
echo "$x + $y = $ans"
```

Floating Point Arithmetic

bash itself can only handle integers. Floating-point solutions include:

bc	basic calculator
calc	has interactive mode
awk	a scripting language
python	a programming language

A Basic Floating-Point Script

```
#!/bin/bash
```

```
read -p "Enter any number: " x  
root=$(echo "sqrt($x)" | bc -l )  
echo "The square root of $x is $root."
```

Conditionals

```
#!/bin/bash
```

```
state=$(cat /sys/class/power_supply/AC/online)
if [ $state = "1" ]; then
    echo "AC"
else echo "BAT"
fi
```

- ▶ Simple conditionals are signalled with `if` and closed with `fi`
- ▶ The condition is contained in brackets (double parentheses if arithmetic)
- ▶ The brackets **must** be spaced: they are a program (`test`)!
- ▶ Action to be undertaken is introduced by `then` or `else`
- ▶ The semicolon allows you to have multiple commands in one line

Conditionals

```
#!/bin/bash
```

Long Form

```
if [ "$(whoami)" != 'root' ]; then  
    echo "Permission denied."  
    exit 1;  
fi
```

Short Form

```
#!/bin/bash
```

```
[ "$(whoami)" != 'root' ] && ( echo "Denied"; exit 1 )
```

- ▶ **&&** for “if successful”; **||** “if unsuccessful”
- ▶ **exit 0** reports success; **exit 1** error; **exit 2** fundamental issues