Uživatel:Zef/output27

Run the last command as root

Useful when you forget to use sudo for a command. "!!" grabs the last run command.

```
$ sudo !!
```

Serve current directory tree at http://$HOSTNAME:8000/

```
$ python -m SimpleHTTPServer
```

Runs previous command but replacing

Really useful for when you have a typo in a previous command. Also, arguments default to empty so if you accidentally run: echo "no typozs" you can correct it with ^z

```
$ ^foo^bar
```

Rapidly invoke an editor to write a long, complex, or tricky command

Next time you are using your shell, try typing ctrl-x e (that is holding control key press x and then e). The shell will take what you've written on the command line thus far and paste it into the editor specified by $EDITOR. Then you can edit at leisure using all the powerful macros and commands of vi, emacs, nano, or whatever. Show Sample Output

```
$ ctrl-x e
```

Place the argument of the most recent command on the shell

When typing out long arguments, such as: cp file.txt /var/www/wp-content/uploads/2009/03/ You can put that argument on your command line by holding down the ALT key and pressing the period '.' or by pressing <ESC> then the period '.'. For example: cd 'ALT+.' would put '/var/www/wp-content/uploads/2009/03/ as my argument. Keeping pressing 'ALT+.' to cycle through arguments of your commands starting from most recent to oldest. This can save a ton of typing.

```
$ 'ALT+.' or '<ESC> .'
```

Salvage a borked terminal

If you bork your terminal by sending binary data to STDOUT or similar, you can get your terminal back using this command rather than killing and restarting the session. Note that you often won't be able to see the characters as you type them. Show Sample Output

```
$ reset
```

currently mounted filesystems in nice layout

Particularly useful if you're mounting different drives, using the following command will allow you to see all the filesystems currently mounted on your computer and their respective specs with the added benefit of nice formatting. Show Sample Output

```
$ mount | column -t
```

Execute a command at a given time

This is an alternative to cron which allows a one-off task to be scheduled for a certain time.

```
$ echo "ls -l" | at midnight
```

Get your external IP address

curl ifconfig.me/ip -> IP Adress curl ifconfig.me/host -> Remote Host curl ifconfig.me/ua ->User Agent curl ifconfig.me/port -> Port thonks to http://ifconfig.me/

```
$ curl ifconfig.me
```

Quick access to the ascii table.

```
$ man ascii
```

Query Wikipedia via console over DNS

Query Wikipedia by issuing a DNS query for a TXT record. The TXT record will also include a short URL to the complete corresponding Wikipedia entry.You can also write a little shell script like:

```
$ cat wikisole.sh #!/bin/sh dig +short txt ${1}.wp.dg.cx and run it like ./wikisole.sh unix were your first option ($1) will be used as search term. Show Sample Output
```

```
$ dig +short txt <keyword>.wp.dg.cx
```

Download an entire website

-p parameter tells wget to include all files, including images. -e robots=off you don't want wget to obey by the robots.txt file -U mozilla as your browsers identity. --random-wait to let wget chose a random number of seconds to wait, avoid get into black list. Other Useful wget Parameters: --limit-rate=20k limits the rate at which it downloads files. -b continues wget after logging out. -o $HOME/wget_log.txt logs the output

```
$ wget --random-wait -r -p -e robots=off -U mozilla http://www.example.com
```

Clear the terminal screen

```
$ ctrl-l
```

A very simple and useful stopwatch

time read -sn1 (s:silent, n:number of characters. Press any character to stop)

```
$ time read (ctrl-d to stop)
```

Put a console clock in top right corner

A nice way to use the console in full screen without forget the current time. you can too add other infos like cpu and mem use. Show Sample Output

```
$ while sleep 1;do tput sc;tput cup 0 $(($(tput cols)-29));date;tput rc;done &
```

List of commands you use most often

```
$ history | awk '{a[$2]++}END{for(i in a){print a[i] " " i}}' | sort -rn | head
```

32 bits or 64 bits?

Easy and direct way to find this out. Show Sample Output

```
$ getconf LONG_BIT
```

Display a block of text with AWK

I find this terribly useful for grepping through a file, looking for just a block of text. There's "grep -A # pattern file.txt" to see a specific number of lines following your pattern, but what if you want to see the whole block? Say, the output of "dmidecode" (as root): dmidecode | awk '/Battery/,/^$/' Will show me everything following the battery block up to the next block of text. Again, I find this extremely useful when I want to see whole blocks of text based on a pattern, and I don't care to see the rest of the data in output. This could be used against the '/etc/securetty/user' file on Unix to find the block of a specific user. It could be used against VirtualHosts or Directories on Apache to find specific definitions. The scenarios go on for any text formatted in a block fashion. Very handy.

```
$ awk '/start_pattern/,/stop_pattern/' file.txt
```

Show File System Hierarchy

Curious about differences between /bin, /usr/bin, and /usr/local/bin? What should be in the /sbin dir? Try this command to find out. Tested against Red Hat & OS X

```
$ man hier
```

Display which distro is installed

```
$ cat /etc/issue
```

A fun thing to do with ram is actually open it up and take a peek. This command will show you all the string (plain text) values in ram

```
$ sudo dd if=/dev/mem | cat | strings
```

Define a quick calculator function

defines a handy function for quick calculations from cli. once defined: ? 10*2+3 Show Sample Output

```
$ ? () { echo "$*" | bc -l; }
```

Graphical tree of sub-directories Prints a graphical directory tree from your current directory Show Sample Output

```
$ ls -R | grep ":$" | sed -e 's/:$//' -e 's/[^-][^\/]*\//--/g' -e 's/^/ /' -e 's/-/|/'
```

Rapidly invoke an editor to write a long, complex, or tricky command

```
$ fc
```

Create a pdf version of a manpage Quick and dirty version. I made a version that checks if a manpage exists (but it's not a oneliner). You must have ps2pdf and of course Ghostscript installed in your box. Enhancements appreciated :-)

```
$ man -t manpage | ps2pdf - filename.pdf
```

Search for a <pattern> string inside all files in the current directory This is how I typically grep. -R recurse into subdirectories, -n show line numbers of matches, -i ignore case, -s suppress "doesn't exist" and "can't read" messages, -I ignore binary files (technically, process them as having no matches, important for showing inverted results with -v) I have grep aliased to "grep --color=auto" as well, but that's a matter of formatting not function.

```
$ grep -RnisI <pattern> *
```

Show numerical values for each of the 256 colors in bash

Same as http://www.commandlinefu.com/commands/view/5876, but for bash. This will show a numerical value for each of the 256 colors in bash. Everything in the command is a bash builtin, so it should run on any platform where bash is installed. Prints one color per line. If someone is interested in formatting the output, paste the alternative.

```
$ for code in {0..255}; do echo -e "\e[38;05;${code}m $code: Test"; done
```

Processor / memory bandwidthd? in GB/s

Read 32GB zero's and throw them away. How fast is your system? Show Sample Output

```
$ dd if=/dev/zero of=/dev/null bs=1M count=32768
```

Makes the permissions of file2 the same as file1

Also works with: chgrp --reference file1 file2 chown --reference file1 file2

```
$ chmod --reference file1 file2
```

List all bash shortcuts

This command shows the various shortcuts that can be use in bash, including Ctrl+L, Ctrl+R, etc... You can translate "\C-y" to Ctrl+y, for example. Show Sample Output

```
$ bind -P
```

Start COMMAND, and kill it if still running after 5 seconds

```
$ timeout 5s COMMAND
```

pretend to be busy in office to enjoy a cup of coffee just make some data scrolling off the terminal. wow.

```
$ cat /dev/urandom | hexdump -C | grep "ca fe"
```

Create a quick back-up copy of a file

Uses shell expansion to create a back-up called file.txt.bak

```
$ cp file.txt{,.bak}
```

Eavesdrop on your system This command takes a snapshot of the open files for a PID 1234 then waits 10 seconds and takes another snapshot of the same PID, it then displays the difference between each snapshot to give you an insight into what the application is doing.

```
$ diff <(lsof -p 1234) <(sleep 10; lsof -p 1234)
```

Nice weather forecast on your shell Change Seville for your prefered city. Show Sample Output

```
$ curl wttr.in/seville
```

Remind yourself to leave in 15 minutes If you spend most of your time in front of the terminal, leave is a useful reminder. Leave can have absolute form: leave 1555 reminds you to leave at 3:55PM Show Sample Output

```
$ leave +15
```

Convert Youtube videos to MP3 youtube-dl has this functionality built in. If you're running an older version of youtube-dl, you can update it using `youtube-dl -U` (although if you have an older version, it probably doesn't download youtube videos anyway.) youtube-dl --help will show you other options that may come in useful.

```
$ youtube-dl -t --extract-audio --audio-format mp3 YOUTUBE_URL_HERE
```

Compare two directory trees. This uses Bash's "process substitution" feature to compare (using diff) the output of two different process pipelines.

```
$ diff <(cd dir1 && find | sort) <(cd dir2 && find | sort)
```

Quick access to ASCII code of a key

```
$ showkey -a
```

Show apps that use internet connection at the moment. show only the name of the apps that are using internet Show Sample Output

```
$ lsof -P -i -n | cut -f 1 -d " "| uniq | tail -n +2
```

Remove blank lines from a file using grep and save output to new file

```
$ grep . filename > newfilename
```

Random Number Between 1 And X

If X is 5, it will about a number between 1 and 5 inclusive. This works in bash and zsh. If you want between 0 and 4, remove the +1.

```
$ echo $[RANDOM%X+1]
```

Prettify an XML file Generated XML files often are poorly formatted. Use this command to properly indent and normalize the file in-place.

```
$ tidy -xml -i -m [file]
```

intersection between two files

```
$ grep -Fx -f file1 file2
```

Colorful man That command installs "most" and make this command as the default man reader. The "most" works like "less" (the current man reader), but it render colors for manpages and may do more things. Read "man most". You can see a preview here: http://www.dicas-l.com.br/dicas-l/20090718.php

```
$ apt-get install most && update-alternatives --set pager /usr/bin/most
```

Create strong, but easy to remember password

Why remember? Generate! Up to 48 chars, works on any unix-like system (NB: BSD use md5 instead of md5sum) Show Sample Output

```
$ read -s pass; echo $pass | md5sum | base64 | cut -c -16
```

Instead of writing a multiline if/then/else/fi construct you can do that by one line

instead of writing: if "$1" == "$2" ; then echo "$1 is equal $2" else echo "$1 differs from $2" fi do write: "$1" == "$2" && echo "$1 is equal $2" || echo "$1 differs from $2"

```
$ [[ test_condition ]] && if_true_do_this || otherwise_do_tha
```

List installed deb packages by size

```
$ dpkg-query -Wf '${Installed-Size}\t${Package}\n' | sort -n
```

find files containing text

-l outputs only the file names -i ignores the case -r descends into subdirectories

```
$ grep -lir "some text" *
```

Fast, built-in pipe-based data sink

This is shorter and actually much faster than >/dev/null (see sample output for timings) Plus, it looks like a disappointed face emoticon. Show Sample Output

```
$ <COMMAND> |:
```

Find Duplicate Files (based on size first, then MD5 hash) If you have the fdupes command, you'll save a lot of typing. It can do recursive searches (-r,-R) and it allows you to interactively select which of the duplicate files found you wish to keep or delete.

```
$ fdupes -r .
```

convert filenames in current directory to lowercase

The simplest way I know. Show Sample Output

```
$ rename 'y/A-Z/a-z/' *
```

Cut out a piece of film from a file. Choose an arbitrary length and starting time.

With: -vcodec, you choose what video codec the new file should be encoded with. Run ffmpeg -formats E to list all available video and audio encoders and file formats. copy, you choose the video encoder that just copies the file. -acodec, you choose what audio codec the new file should be encoded with. copy, you choose the audio encoder

that just copies the file. -i originalfile, you provide the filename of the original file to ffmpeg -ss 00:01:30, you choose the starting time on the original file in this case 1 min and 30 seconds into the film -t 0:0:20, you choose the length of the new film newfile, you choose the name of the file created. Here is more information of how to use ffmpeg: http://www.ffmpeg.org/ffmpeg-doc.html

```
$ ffmpeg -i input.mp4 -ss 00:08:32 -to 00:09:32 -c copy output.mp4
```

perl one-liner to get the current week number

Not perl but shorter.

```
$ date +%V
```

Using awk to sum/count a column of numbers.

Takes a input file (count.txt) that looks like: 1 2 3 4 5 It will add/sum the first column of numbers.

```
$ cat count.txt | awk '{ sum+=$1} END {print sum}'
```

Clean up poorly named TV shows.

Replace 'SHOWNAME' with the name of the TV show. Add -n to test the command without renaming files. Check the 'sample output'. Show Sample Output

```
$ rename -v 's/.*[s,S](\d{2}).*[e,E](\d{2}).*\.avi/SHOWNAME\ S$1E$2.avi/' poorly.named.file.s01e01.avi
```