What is your single most favorite command-line trick using Bash? [closed]

Asked 16 years, 3 months ago Modified 13 years, 2 months ago

Viewed 148k times

156

votes

It's difficult to tell what is being asked here. This question is ambiguous, vague, incomplete, overly broad, or rhetorical and cannot be reasonably answered in its current form. For help clarifying this question so that it can be reopened, visit the help center.

Closed 13 years ago.

Locked. This question and its answers are locked because the question is off-topic but has historical significance. It is not currently accepting new answers or interactions.

We all know how to use `<ctrl>-R` to reverse search through history, but did you know you can use `<ctrl>-S` to forward search if you set `stty stop ""`? Also, have you ever tried running bind -p to see all of your keyboard shortcuts listed? There are over 455 on Mac OS X by default.

What is your single most favorite obscure trick, keyboard shortcut or shopt configuration using bash?

bash command-line

Share

edited Oct 5, 2011 at 3:09

community wiki
11 revs, 6 users 62%
Robert Harvey

2 Please reword this to say "What is your single most favourite". This allows people to up-vote specific answers, almost like a poll. – SCdF Sep 16, 2008 at 1:08

> Please reword this to say "What is your single most favourite". Done. – Martin Klinke Oct 2, 2008 at 22:05

1 There is a StackOverflow clone for this very question: commandlinefu.com – rkb Apr 15, 2009 at 16:54

Only 232 of those 455 default key-bindings do something other than "self-insert" ("type this key"): $ bind -p |grep -v self-insert | wc – Ed Brannin Jun 2, 2009 at 10:57

Some neat stuff in here. But it should be noted that a quite a few of them only work when the bash is in emacs mode... – Mo. Sep 10, 2009 at 16:03

Comments disabled on deleted / locked posts / reviews |

105 Answers Sorted by: Highest score (default)

156 votes

Renaming/moving files with suffixes quickly:

```
cp /home/foo/reallylongname.cpp{,-old}
```

This expands to:

```
cp /home/foo/reallylongname.cpp
/home/foo/reallylongname.cpp-old
```

Share

answered Sep 16, 2008 at 1:38

community wiki
user10765

9 Just to point out that to do the reverse (going from .cpp-old to .cpp) you'd do cp /home/foo/reallylongname.cpp{-old,} – Michael Ekoka Sep 3, 2010 at 16:03

149 votes

```
cd -
```

It's the command-line equivalent of the back button (takes you to the previous directory you were in).

Share

answered Sep 16, 2008 at 1:01

community wiki
Mark Biek

6 I prefer to use `pushd` and `popd` to maintain a directory stack, myself. – Alex M Sep 16, 2008 at 1:37

21 But 'cd -' has the advantage of working even if you didn't remember to use pushd. – Sergio Acosta Sep 16, 2008 at 5:50

10 It's worth mentioning that 'cd' takes you to your home directory. – dr-jan Sep 19, 2008 at 19:14

It's interesting that this doesn't show up in `man cd` or, `cd --help` (at least mine). – Halil Özgür Feb 22, 2011 at 16:55

135 votes

Another favorite:

```
!!
```

Repeats your last command. Most useful in the form:

```
sudo !!
```

Share

answered Sep 16, 2008 at 1:07

community wiki
amrox

81 it has the added benefit of making you sound really angry about it too. "Computer, do this." "Access denied". "DO IT!!"

– nickf Sep 16, 2008 at 1:16

That's not little known! :-) – hoyhoy Sep 16, 2008 at 2:00

10 Similar things you can do: `mkdir testdir; cd !$` .. This runs `cd [last word of previous line]` (in the example, `cd testdir`) – dbr Sep 16, 2008 at 10:07

13 Make Me A Sandwich Sudo !! – Kibbee Apr 15, 2009 at 16:45

If you like !!, !$ and !-1:^foo^bar, you'll also "bind Space:magic-space". stackoverflow.com/questions/603696/… – Yoo Sep 13, 2009 at 16:37

80

votes

My favorite is '^string^string2' which takes the last command, replaces string with string2 and executes it

```
$ ehco foo bar baz
bash: ehco: command not found
$ ^ehco^echo
foo bar baz
```

Bash command line history guide

Share

edited May 24, 2011 at 11:52

community wiki
3 revs, 3 users 92%
seth

This is non-greedy. How do I make it greedy? – Altreus Nov 7, 2008 at 11:21

9 `!!:gs/ehco/echo/` – andre-r Sep 26, 2009 at 12:10

The 2nd page of that guide is essential – jmoz Apr 16, 2010 at 10:03

1 @andre-r: `!!:gs/ehco/echo/` performs a global search and replace on the last command (confer `!!` in stackoverflow.com/questions/68372/…). That is not equivalent to `^ehco^echo` which just replaces one instance of "ehco" -- a more accurate response would be `!!:s/ehco/scho`.
– Iceland_jack Jul 30, 2010 at 19:29

---

64 votes

rename

Example:

```
$ ls
this_has_text_to_find_1.txt
this_has_text_to_find_2.txt
this_has_text_to_find_3.txt
this_has_text_to_find_4.txt

$ rename 's/text_to_find/been_renamed/' *.txt
$ ls
this_has_been_renamed_1.txt
this_has_been_renamed_2.txt
this_has_been_renamed_3.txt
this_has_been_renamed_4.txt
```

So useful

Share

answered Sep 16, 2008 at 1:06

community wiki
Jiaaro

3 Wow, I've been a moron all these years using basename, mv and {} tricks to do this. /foreheadsmack. – Gregg Lind Oct 23, 2008 at 21:13

7 rename isn't bash/readline specific like the other posts however. – guns Mar 12, 2009 at 14:49

1 zmv from zsh distribution is much better in most cases. – ZyX May 24, 2010 at 6:07

util-linux-ng has a rename command too and it's not like the one mentioned in this answer. The command described here is actually mmv. – Cristian Ciupitu Nov 17, 2010 at 14:14

60 votes

I'm a fan of the `!$`, `!^` and `!*` expandos, returning, from the most recent submitted command line: the last item, first non-command item, and all non-command items. To wit (Note that the shell prints out the command first):

```
$ echo foo bar baz
foo bar baz
$ echo bang-dollar: !$ bang-hat: !^ bang-star: !*
echo bang-dollar: baz bang-hat: foo bang-star: foo bar b
bang-dollar: baz bang-hat: foo bang-star: foo bar baz
```

This comes in handy when you, say `ls filea fileb`, and want to edit one of them: `vi !$` or both of them: `vimdiff !*`. It can also be generalized to "the `n`th argument" like so:

```
$ echo foo bar baz
$ echo !:2
echo bar
bar
```

Finally, with pathnames, you can get at parts of the path by appending `:h` and `:t` to any of the above expandos:

```
$ ls /usr/bin/id
/usr/bin/id
$ echo Head: !$:h  Tail: !$:t
echo Head: /usr/bin Tail: id
Head: /usr/bin Tail: id
```

Share

edited Sep 18, 2008 at 0:04

community wiki
3 revs
Pi.

the 'echo !!2' didn't work for me and from a later post I saw that I think it's supposed to be: 'echo !:2' – user13060 Sep 17, 2008 at 7:17

8 add "bind Space:magic-space" to .bashrc and any ! combination will be automatically expanded when you hit space. – Yoo Sep 13, 2009 at 16:48

47 votes

When running commands, sometimes I'll want to run a command with the previous ones arguments. To do that, you can use this shortcut:

```
$ mkdir /tmp/new
$ cd !!:*
```

Occasionally, in lieu of using find, I'll break-out a one-line loop if I need to run a bunch of commands on a list of files.

```
for file in *.wav; do lame "$file" "$(basename "$file"
```

Configuring the command-line history options in my .bash_login (or .bashrc) is really useful. The following is a cadre of settings that I use on my Macbook Pro.

Setting the following makes bash erase duplicate commands in your history:

```
export HISTCONTROL="erasedups:ignoreboth"
```

I also jack my history size up pretty high too. Why not? It doesn't seem to slow anything down on today's microprocessors.

```
export HISTFILESIZE=500000
export HISTSIZE=100000
```

Another thing that I do is ignore some commands from my history. No need to remember the exit command.

```
export HISTIGNORE="&:[ ]*:exit"
```

You definitely want to set histappend. Otherwise, bash overwrites your history when you exit.

```
shopt -s histappend
```

Another option that I use is cmdhist. This lets you save multi-line commands to the history as one command.

```
shopt -s cmdhist
```

Finally, on Mac OS X (if you're not using vi mode), you'll want to reset <CTRL>-S from being scroll stop. This prevents bash from being able to interpret it as forward search.

```
stty stop ""
```

Share

edited Oct 3, 2010 at 9:34

community wiki
4 revs, 2 users 98%
hoyhoy

6 I find "Alt-." much better than "!!:*" for repeating the last word of the last command. – Weidenrinde Sep 17, 2008 at 23:34

Additions for histroy: 1. regularly save history (with -a if you have multiple shells open at once) export PROMPT_COMMAND="history -a" 2. Alias for reading the

history from other shells. alias -- h='history -n ; history | grep '
– Weidenrinde Sep 17, 2008 at 23:35

export HISTIGNORE="[ ]*" is not necessary, as your HISTCONTROL=ignoreboth should do the same
– Weidenrinde Sep 17, 2008 at 23:36

Another nice thing for history: export HISTTIMEFORMAT="%Y-%m-%d %H:%M:%S : "
– Weidenrinde Sep 17, 2008 at 23:37

To save a few keystrokes off of "cd !!:*", try "cd !$". The "!$" will be interpreted as the last argument of your last command.
– Tim Stewart Dec 7, 2008 at 0:08

43

votes

How to list *only* subdirectories in the current one ?

```
ls -d */
```

It's a simple trick, but you wouldn't know how much time I needed to find that one !

Share

answered Oct 5, 2008 at 13:02

community wiki
edomaur

2 Excellent! All these years the best I could come up with was `alias lsd='ls -F | grep --color /'`, which would list the same thing but in a more lame fashion. However, it would list one dir per line, for ease of parsing. I've modified your

command to do the same: `ls -d1 */` – Artem Russakovskii Aug 4, 2009 at 15:06

how does this work? "ls -d" lists just . (which is weird) but "ls -d */" works. – Yoo Sep 13, 2009 at 16:58

1 It is a bit tricky... "ls -d" is similar to "ls -d ./" and "ls -d */" to "ls -d ./*/". The '-d' switch set 'ls' tu list only directory entries, but if you give it no parameter, it use the current directory as a parameter, so it has *only* the "." directory to list... – edomaur Sep 14, 2009 at 22:54

1 I can't believe I've never discovered this before. Great tip. – Nick Dixon Jan 20, 2010 at 10:29

No words, just wow!!! :) – dimba Sep 10, 2011 at 7:27

41

votes

ESC .

Inserts the last arguments from your last bash command. It comes in handy more than you think.

```
cp file /to/some/long/path
```

cd ESC .

Share

edited May 24, 2011 at 12:20

community wiki
3 revs, 3 users 63%
amrox

This works for me (bash). Note that you hit the esc key and then the period key. No hyphen or anything. – Howler Sep 16, 2008 at 7:20

9 Also alt-. is the same thing. – Mark Baker Oct 27, 2008 at 11:17

Yeah, it's Bash only. But <esc> (or Alt- or Meta-) _ is bound to the same function, and <esc>-_ also works in vi-mode too. – Nick Dixon Jan 20, 2010 at 10:26

@hendry in my zsh it actually works. Maybe you use vi mode? – ZyX May 24, 2010 at 6:05

Use the kbd-tag for keyboard hits: <kbd>ESC</kbd> (But doesn't play very nice together with code-formatting/indentation). – user unknown May 24, 2011 at 12:17

36 votes

Sure, you can "`diff file1.txt file2.txt`", but Bash supports process substitution, which allows you to `diff` the output of commands.

For example, let's say I want to make sure my script gives me the output I expect. I can just wrap my script in <( ) and feed it to `diff` to get a quick and dirty unit test:

```
$ cat myscript.sh
#!/bin/sh
echo -e "one\nthree"
$
$ ./myscript.sh
one
three
$
$ cat expected_output.txt
one
```

```
two
three
$
$ diff <(./myscript.sh) expected_output.txt
1a2
> two
$
```

As another example, let's say I want to check if two servers have the same list of RPMs installed. Rather than sshing to each server, writing each list of RPMs to separate files, and doing a `diff` on those files, I can just do the `diff` from my workstation:

```
$ diff <(ssh server1 'rpm -qa | sort') <(ssh server2 'rp
241c240
< kernel-2.6.18-92.1.6.el5
---
> kernel-2.6.18-92.el5
317d315
< libsmi-0.4.5-2.el5
727,728d724
< wireshark-0.99.7-1.el5
< wireshark-gnome-0.99.7-1.el5
$
```

There are more examples in the Advanced Bash-Scripting Guide at http://tldp.org/LDP/abs/html/process-sub.html.

Share

edited Oct 2, 2008 at 22:05

community wiki
2 revs
Philip Durbin

35

votes

My favorite command is "ls -thor"

It summons the power of the gods to list the most recently modified files in a conveniently readable format.

Share

answered Sep 16, 2008 at 4:55

community wiki
user10109

6 Reminds me of another, totally useless but also funny command: `ls -bart -simpson -ruls`. S, ICNR – filiprem Apr 24, 2010 at 0:49

'ls -tor' is also useful since you can quickly find large files. And also, the original Swedish spelling of the name of the god of thunder is actually 'Tor'. – dala May 17, 2010 at 12:41

In Danish you can use `ls -lort` which is the Danish word for "shit". Omitting any groups and not getting the wrath of valhal :) – Jesper Rønn-Jensen May 28, 2010 at 21:01

1 one could also say it like "ls -lotr" which apparently has something to do with Tolkien and Lord Of The Rings :) – ADEpt Aug 7, 2010 at 17:37

28

votes

More of a novelty, but it's clever...

Top 10 commands used:

```
$ history | awk '{print $2}' | awk 'BEGIN {FS="|"}{print
| sort -nr | head
```

Sample output:

```
 242 git
  83 rake
  43 cd
  33 ss
  24 ls
  15 rsg
  11 cap
  10 dig
   9 ping
   3 vi
```

Share

edited Oct 3, 2010 at 9:32

community wiki
3 revs, 3 users 82%
ctcherry

1 Here's a shorter, faster version: `history | awk 'BEGIN {FS="[ \t]+|\\|"} {print $3}' | sort | uniq -c | sort -nr | head` – Dennis Williamson Jun 10, 2010 at 4:17

25

votes

^R reverse search. Hit ^R, type a fragment of a previous command you want to match, and hit ^R until you find the one you want. Then I don't have to remember recently used commands that are still in my history. Not exclusively bash, but also: ^E for end of line, ^A for beginning of line, ^U and ^K to delete before and after the cursor, respectively.

Share

answered Sep 16, 2008 at 1:33

community wiki
user10741

2 I can never remember ^U for some reason. Isn't there a word delete delete shortcut too? – hoyhoy Sep 16, 2008 at 1:36

1 ^W deletes the word before the cursor. (Feel free to edit your answer with this.) – Jon Ericson Sep 16, 2008 at 6:22

2 ^R being so useful that it was mentioned in the question :) – Mark Baker Oct 27, 2008 at 11:21

20

votes

I often have aliases for vi, ls, etc. but sometimes you want to escape the alias. Just add a back slash to the command in front:

Eg:

```
$ alias vi=vim
$ # To escape the alias for vi:
$ \vi # This doesn't open VIM
```

Cool, isn't it?

Share

answered Sep 16, 2008 at 11:12

community wiki
Srikanth

> You can also run `command` —see `SHELL BUILTIN COMMANDS` in the Bash manual. – Iceland_jack Jan 2, 2011 at 20:00

17

votes

Here's a couple of configuration tweaks:

`~/.inputrc`:

```
"\C-[[A": history-search-backward
"\C-[[B": history-search-forward
```

This works the same as `^R` but using the arrow keys instead. This means I can type (e.g.) `cd /media/` then hit up-arrow to go to the last thing I `cd`'d to inside the `/media/` folder.

(I use Gnome Terminal, you may need to change the escape codes for other terminal emulators.)

Bash completion is also incredibly useful, but it's a far more subtle addition. In `~/.bashrc`:

```
if [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
fi
```

This will enable per-program tab-completion (e.g. attempting tab completion when the command line starts with `evince` will only show files that evince can open, and it will also tab-complete command line options).

Works nicely with this also in `~/.inputrc`:

```
set completion-ignore-case on
set show-all-if-ambiguous on
set show-all-if-unmodified on
```

Share

edited Sep 16, 2008 at 1:19

community wiki
2 revs
Porges

+1 bash_completion is awesome – prestomation Jan 15, 2010 at 20:34

On a Mac, the History search with arrow keys can be enabled with: bind '"\e[A":history-search-backward bind '"\e[B":history-search-forward – Asmus Feb 2, 2011 at 12:42

17 votes

I use the following a lot:

The `:p` modifier to print a history result. E.g.

```
!!:p
```

Will print the last command so you can check that it's correct before running it again. Just enter `!!` to execute it.

In a similar vein:

```
!?foo?:p
```

Will search your history for the most recent command that contained the string 'foo' and print it.

If you don't need to print,

```
!?foo
```

does the search and executes it straight away.

Share

edited Sep 17, 2008 at 18:03

community wiki
2 revs
Steve Lacey

2 Instead of using :p, you can also use magic-space: stackoverflow.com/questions/603696/… – Yoo Sep 13, 2009 at 17:38

to expand the historical controls without running confiure magick-space in bashrc `bind Space:magic-space # !pin<space>` – SergioAraujo Oct 11, 2011 at 19:00

16

votes

I have got a secret weapon : shell-fu.

There are thousand of smart tips, cool tricks and efficient recipes that most of the time fit on a single line.

One that I love (but I cheat a bit since I use the fact that Python is installed on most Unix system now) :

```
alias webshare='python -m SimpleHTTPServer'
```

Now everytime you type "webshare", the current directory will be available through the port 8000. Really nice when you want to share files with friends on a local network without usb key or remote dir. Streaming video and music will work too.

And of course the classic fork bomb that is completely useless but still a lot of fun :

```
$ :(){ :|:& };:
```

Don't try that in a production server...

Share

edited May 28, 2010 at 21:39

community wiki
2 revs, 2 users 97%
e-satis

I guess i shouldn't webshare / – Yoo Sep 13, 2009 at 17:42

I guess you can simplify the command to `python -m SimpleHTTPServer` (unless I'm missing something fancy the original command does – Jesper Rønn-Jensen May 28, 2010 at 21:38

Meh. On a production server that fork bomb will just get squelched by the OS; there are limits on the number of processes that any user can create. – Donal Fellows Aug 9, 2010 at 2:11

3 @Donal: You should try it and report your findings. – Michael F Sep 16, 2010 at 7:04

12 votes

You can use the watch command in conjunction with another command to look for changes. An example of this was when I was testing my router, and I wanted to get up-to-date numbers on stuff like signal-to-noise ratio, etc.

```
watch --interval=10 lynx -dump http://dslrouter/stats.ht
```

Share

answered Sep 16, 2008 at 0:59

community wiki
HFLW

The watch command is really cool. I first read about in the Linux Server Hacks book about five years ago. Who knew it existed? – hoyhoy Sep 16, 2008 at 1:34

11 What in the world does 'watch' have to do with bash? – Artem Russakovskii Aug 4, 2009 at 14:59

11 votes

```
type -a PROG
```

in order to find all the places where PROG is available, usually somewhere in ~/bin rather than the one in /usr/bin/PROG that might have been expected.

Share

answered Sep 16, 2008 at 6:07

community wiki
dajobe

1 It will also tell you if it's a function or alias. If it's in multiple places in your `PATH` it will show each of them. – Dennis Williamson Jun 10, 2010 at 4:25

11 votes

I like to construct commands with **echo** and pipe them to the shell:

```
$ find dir -name \*~ | xargs echo rm
...
$ find dir -name \*~ | xargs echo rm | ksh -s
```

Why? Because it allows me to look at what's going to be done before I do it. That way if I have a horrible error (like removing my home directory), I can catch it before it happens. Obviously, this is most important for destructive or irrevocable actions.

Share

edited Sep 16, 2008 at 6:19

community wiki
2 revs
Jon Ericson

1 You might want to use find -print0 and xargs -0 to avoid problems with files and folders with white spaces in them. – neu242 Sep 22, 2008 at 8:13

Or avoid creating and working with files and folders with white spaces in their names. ;-) Obviously if you are working with user generated names, your suggestion is vital. – Jon Ericson Sep 22, 2008 at 16:09

4 Why would you append "| ksh -s" when you could just remove the "echo"? – Yoo Sep 13, 2009 at 17:27

No need to pipe with GNU find: `$ find dir -name \\*~ -exec rm {} +` – bobbogo Jan 24, 2011 at 12:01

10 votes

When downloading a large file I quite often do:

```
while ls -la <filename>; do sleep 5; done
```

And then just ctrl+c when I'm done (or if `ls` returns non-zero). It's similar to the `watch` program but it uses the shell instead, so it works on platforms without `watch`.

Another useful tool is netcat, or `nc`. If you do:

```
nc -l -p 9100 > printjob.prn
```

Then you can set up a printer on another computer but instead use the IP address of the computer running netcat. When the print job is sent, it is received by the computer running netcat and dumped into `printjob.prn`.

Share

edited Sep 16, 2008 at 1:13

community wiki
2 revs
dreamlax

Why this and not, for example, `wget`? – porges Sep 16, 2008 at 1:12

Try `watch ls -la <filename>` (use -n5 if you don't like the default of 2 seconds). – T Percival Sep 16, 2008 at 4:46

1 @Ted, `watch` is not available on all platforms though, like Solaris, Mac OS X, etc. @Porges, not sure what you mean. Can wget also listen on a port? – dreamlax Sep 16, 2008 at 21:57

`pushd` and `popd` almost always come in handy

10

votes

Share

answered Sep 16, 2008 at 2:07

community wiki
jdt141

10

votes

One preferred way of navigating when I'm using multiple directories in widely separate places in a tree hierarchy is to use acf_func.sh (listed below). Once defined, you can do

cd --

to see a list of recent directories, with a numerical menu

cd -2

to go to the second-most recent directory.

Very easy to use, very handy.

Here's the code:

```
# do ". acd_func.sh"
# acd_func 1.0.5, 10-nov-2004
# petar marinov, http:/geocities.com/h2428, this is publ

cd_func ()
{
  local x2 the_new_dir adir index
  local -i cnt

  if [[ $1 ==  "--" ]]; then
    dirs -v
    return 0
```

```
  fi

  the_new_dir=$1
  [[ -z $1 ]] && the_new_dir=$HOME

  if [[ ${the_new_dir:0:1} == '-' ]]; then
    #
    # Extract dir N from dirs
    index=${the_new_dir:1}
    [[ -z $index ]] && index=1
    adir=$(dirs +$index)
    [[ -z $adir ]] && return 1
    the_new_dir=$adir
  fi

  #
  # '~' has to be substituted by ${HOME}
  [[ ${the_new_dir:0:1} == '~' ]] && the_new_dir="${HOME

  #
  # Now change to the new dir and add to the top of the
  pushd "${the_new_dir}" > /dev/null
  [[ $? -ne 0 ]] && return 1
  the_new_dir=$(pwd)

  #
  # Trim down everything beyond 11th entry
  popd -n +11 2>/dev/null 1>/dev/null

  #
  # Remove any other occurence of this dir, skipping the
  for ((cnt=1; cnt <= 10; cnt++)); do
    x2=$(dirs +${cnt} 2>/dev/null)
    [[ $? -ne 0 ]] && return 0
    [[ ${x2:0:1} == '~' ]] && x2="${HOME}${x2:1}"
    if [[ "${x2}" == "${the_new_dir}" ]]; then
      popd -n +$cnt 2>/dev/null 1>/dev/null
      cnt=cnt-1
    fi
  done

  return 0
}
```

```
alias cd=cd_func

if [[ $BASH_VERSION > "2.05a" ]]; then
  # ctrl+w shows the menu
  bind -x "\"\C-w\":cd_func -- ;"
fi
```

Share

answered Sep 16, 2008 at 3:07

community wiki
Leonard

---

10 votes

Expand complicated lines before hitting the dreaded enter

- Alt + Ctrl + e — *shell-expand-line* (may need to use Esc , Ctrl + e on your keyboard)
- Ctrl + _ — *undo*
- Ctrl + x , * — *glob-expand-word*

$ `echo !$ !-2^ *` Alt + Ctrl + e
$ `echo aword someotherword *` Ctrl + _
$ `echo !$ !-2^ *` Ctrl + x , *
$ `echo !$ !-2^ LOG Makefile bar.c foo.h`

&c.

Share

answered Jan 6, 2011 at 13:50

community wiki
bobbogo

+1, I am always afraid of pressing `Enter` :) – João Portela Jan 6, 2011 at 14:30

Didn't know any of these. Undo is a killer. Good for any command line editing. – Philippe A. Jan 30, 2012 at 17:36

9

votes

I've always been partial to:

```
ctrl-E # move cursor to end of line
ctrl-A # move cursor to beginning of line
```

I also use `shopt -s cdable_vars`, then you can create bash variables to common directories. So, for my company's source tree, I create a bunch of variables like:

```
export Dcentmain="/var/localdata/p4ws/centaur/main/apps/
```

then I can change to that directory by `cd Dcentmain`.

Share

answered Sep 16, 2008 at 3:38

community wiki
Drew Frezell

Home and End do the same things as crtl-A and ctrl-E. – davidfg4 Apr 15, 2009 at 17:01

This helps on Mac Laptops that don't have home and end keys. – jamesh Dec 4, 2009 at 10:04

Unfortunately ctrl-A is also the escape key for `screen` by default. I remap it to ctrl-X in my `.screenrc`, but then I'm not an emacs user. – system PAUSE Dec 11, 2009 at 21:44

Somewhat related to the cdable_vars option, there's also the CDPATH environment variable. But that works differently, because you set the parent directory, much like in PATH. With cdable_vars you get more control. – Zecc Nov 17, 2010 at 10:45

1 More for your shortcut list: CTRL+K to delete everything from cursor to end of line, CTRL+U to delete everything before cursor, ALT+F/ALT+B to move one word forward/backward (or CTRL+left arrow/CTRL+right arrow). +1 for cdable_vars! – Philippe A. Jan 30, 2012 at 17:39

8

votes

`pbcopy`

This copies to the Mac system clipboard. You can pipe commands to it...try:

```
pwd | pbcopy
```

Share

answered Oct 5, 2009 at 15:32

community wiki
Max Masnick

N.B! This is a Mac specific command. – dala May 17, 2010 at 13:33

These aren't: `alias pbcopy='xsel --clipboard --input'` and `alias pbpaste='xsel --clipboard --output'` – wawawawa Oct 18, 2011 at 8:45

7

votes

```
$ touch {1,2}.txt
$ ls [12].txt
1.txt  2.txt
$ rm !:1
rm [12].txt
$ history | tail -10
...
10007  touch {1,2}.txt
...
$ !10007
touch {1,2}.txt
$ for f in *.txt; do mv $f ${f/txt/doc}; done
```

Share

answered Sep 16, 2008 at 1:28

community wiki
user9706

Use `history 10` instead of `history | tail -10` (which should be `tail -n 10` by the way since that syntax is deprecated). – Dennis Williamson Jun 10, 2010 at 4:24

7

votes

Using 'set -o vi' from the command line, or better, in .bashrc, puts you in vi editing mode on the command line. You start in 'insert' mode so you can type and backspace as normal, but if you make a 'large' mistake you can hit the

esc key and then use 'b' and 'f' to move around as you do in vi. cw to change a word. Particularly useful after you've brought up a history command that you want to change.

Share

answered Sep 16, 2008 at 2:52

community wiki
Leonard

---

7 votes

Similar to many above, my current favorite is the keystroke [alt]. (Alt and "." keys together) this is the same as $! (Inserts the last argument from the previous command) except that it's immediate and for me easier to type. (Just can't be used in scripts)

eg:

```
mkdir -p /tmp/test/blah/oops/something
cd [alt].
```

Share

answered Sep 16, 2008 at 8:58

community wiki
user11535

Also try combining it with Alt-[0-9] (hit the first combination, then release, then the second). e.g. if last command was 'mv foo bar', then "Alt-0 Alt-." gives 'mv', "Alt-1 Alt-." gives 'foo', "Alt-2

Alt-." and basic Alt-. both give 'bar', etc. – Sam Stokes Sep 24, 2008 at 23:39

1 Also try pressing Alt-. more than once (or press down Alt, hold it there, press dot many times, and then release Alt) This is similar to pressing Ctrl-R more than once. – Yoo Sep 13, 2009 at 17:34

7

votes

String multiple commands together using the **&&** command:

```
./run.sh && tail -f log.txt
```

or

```
kill -9 1111 && ./start.sh
```

Share

edited Mar 18, 2010 at 10:42

community wiki
2 revs, 2 users 80%
TChen

1 if one fail, the other command wont be executed (fail fast, like in programmation) – Frederic Morin Apr 17, 2009 at 9:27

1 The opposite is to use the logical or '||' in which the right hand side will only be executed if the left hand side is false. Example: <code>command_1 || command_2</code> – dala May 17, 2010 at 13:26