Bash History Cheat Sheet

Emacs and Vi history editing keyboard shortcuts:

Shortcut	Description
Emacs Mode Shortcuts:	
CTRL-p	Fetch the previous command from the history list.
CTRL-n	Fetch the next command from the history list.
CTRL-r	Search history backward (incremental search).
CTRL-s	Search history forward (incremental search).
Meta-p	Search backward using non-incremental search.
Meta-n	Search forward using non-incremental search.
Meta-<	Move to the first line in the history.
Meta->	Move to the end of the history list.
Vi Mode Shortcuts:	
k	Fetch the previous command from the history list.
j	Fetch the next command from the history list.
/*string* or **CTRL-r**	Search history backward for a command matching *string*.
?*string* or **CTRL-s**	Search history forward for a command matching *string*.
n	Repeat search in the same direction as previous.
N	Repeat search in the opposite direction as previous.
G	Move to the N-th history line (for example, **15G**).

History behavior modification via shell variables:

Shell Variable	Description
HISTFILE	Controls where the history file gets saved. Set to **/dev/null** not to keep history. Default: ~/.bash_history.
HISTFILESIZE	Controls how many history commands to keep in **HISTFILE**. Default: 500.
HISTSIZE	Controls how many history commands to keep in the history list of current session. Default: 500.
HISTIGNORE	Controls which commands to ignore and not save to the history list. The variable takes a list of colon separated patterns. Pattern **&** matches the previous history command.

History behavior modification via *shopt* command:

shopt option	Description
histappend	Setting the variable appends current session history to **HISTFILE**. Unsetting overwrites the file each time.
histreedit	If set, puts a failed history substitution back on the command line for re-editing.
histverify	If set, puts the command to be executed after a substitution on command line as if you had typed it.

shopt options can be set by a `shopt -s option` and can be unset by a `shopt -u option` shell command.

A cheat sheet by **Peter Krumins** (peter@catonmat.net, @pkrumins on twitter)
www.catonmat.net – good coders code, great coders reuse

History expansion:

Shortcut	Description
Event Designators:	
!	Starts a history substitution.
!!	Refers to the last command.
!n	Refers to the **n**-th command line.
!-n	Refers to the current command line minus **n**.
!string	Refers to the most recent command starting with **string**.
!?string?	Refers to the most recent command containing **string** (the ending ? is optional).
^string1^string2^	Quick substitution. Repeats the last command, replacing **string1** with **string2**.
!#	Refers to the entire command line typed so far.
Word Designators (word designators follow the event designators, separated by a colon):	
0	The zeroth (first) word in a line (usually command name).
n	The **n**-th word in a line.
^	The first argument (the second word) in a line.
$	The last argument in a line.
%	The word matched by the most recent ?string? search.
x-y	A range of words from **x** to **y** (**-y** is synonymous with **0-y**).
*****	All word but the zeroth.
x*	Synonymous with **x-$**.
x-	The words from **x** to the second to last word.
Modifiers (modifiers follow word designators, separated by a colon):	
h	Removes a trailing pathname component, leaving the head.
t	Removes all leading pathname components, leaving the tail.
r	Removes a trailing suffix of the form .xxx, leaving the basename.
e	Removes all but the trailing suffix.
p	Prints the resulting command but does not execute it.
q	Quotes the substituted words, escaping further substitutions.
x	Quotes the substituted words, breaking them into words at blanks and newlines.
s/old/new/	Substitutes **new** for **old**.
&	Repeats the previous substitution.
g	Causes **s/old/new/** or **&** to be applied over the entire event line.

History expansion examples:

```
$ echo a b c d e    (executes ‘echo ab c d e‘)
a b c d e
$ echo !!:3-$       (executes ‘echo c d e‘)
c d e
$ echo !-2:*:q      (executes ‘echo ’a b c d e’‘)
a b c d e
$ echo !-3:1:2:4:x  (executes ‘echo ’a’ ’b’ ’d’‘)
a b d
$ echo !-4:1-3:s/a/foo/:s/b/bar/:s/c/baz/
(executes ‘echo foo bar baz‘)
foo bar baz
```

```
$ tar -xzf package-x.y.z.tgz
...
$ cd !-1:$:r  (exec’s ‘cd package-x.y.z‘)
package-x.y.z $

$ ls -a /tmp
file1  file2  file3 ...
$ ^-a^-l^     (exec’s ‘ls -l /tmp‘)
-rw-------  1  user  user  file1
...
```

A cheat sheet by **Peter Krumins** (peter@catonmat.net, @pkrumins on twitter)
www.catonmat.net – good coders code, great coders reuse