

# Bash Binds

The completion keybinds use mnemonic symbols for each category. @ for hostname, / pathsep for files, ! bang for commands. Known hosts won't complete if they are hashed, which they probably should be. Most of Bash syntax has similar mnemonics in the syntax for keys & string expansions, but it sounds a bit weird explaining it.

## Emacs Bindings

abort C-g, C-C-g, M-C-g	<b>digit</b>	<b>insert</b>	spell-correct-word C-x s	abort C-g	digit-argument M [0-9-]
accept-line C-j, C-m	argument M--	comment M-#	start-kbd-macro C-x (	capitalize-word M-c	downcase-word M-l
<b>backward</b>	argument M-0	completions M-*	tilde-expand M-&	character-search C ] \$char	dynamic-complete-history M c i
char C-b, <left>	argument M-1	last-argument M--, M_-		backward M C ] \$char	edit-and-execute-command C xC e
delete C-h	argument M-2				exchange-point-and-mark C xC x
<b>kill</b>	argument M-3				
line C-x C-?	argument M-4	<b>kill</b>			
word M-c, M-?<left>	argument M-5	line C-k			
word M-b, M-?<left>	argument M-6	word C->delete>, M-d			
<b>beginning-of</b>	argument M-7				
history M-!, <pgup>	argument M-8				
line C-a, <home>	argument M-9	<b>next</b>			
display-shell-version C-x C-v		history C-n, M-<down>			
dynamic-complete-history M-c i		<b>non-incremental</b>			
edit-and-execute-command C x C-e		forward-search-history M-n			
<b>character-search C-!</b>		reverse-search-history M-p			
backward M-C-] \$char		<b>operate-and-get-next</b> C-o			
<b>clear</b>		unix			
display M-C-l		line-discard C-u			
screen C-l		word-rubout C-w			
<b>complete</b> C-i, <esc> <esc>		<b>upcase-word</b> M-u			
command M-!		<b>vank</b> C-y			
filename M-		last-arg M--, M_-			
hostname M-@		nth-arg M-C-y			
into-braces M-{		pop M-y			
username M-~		<b>possible</b>			
variable M-\$		command-completions C-x!			
<b>delete</b>		completions M-, M-?			
char C-d, <delete>		filename-completions C-x@			
horizontal-space M-\`		hostname-completions C-x@			
		username-completions C-x-			
		variable-completions C-x\$			
		<b>previous</b>			
		history C-p, M-<up>			
		<b>forward</b>			
		history C-p, M-<down>			
		re-read-init-file C-x C-r			
		reverse-search-history C-r			
		revert-line M-C-r, M-r			
		set-mark C-@, M-"			
		<b>glob</b>			
		complete-word M-g			
		expand-word C-x*			
		list-expansions C-xg			
		<b>shell</b>			
		backward-word M-C-b			
		expand-line M-C-e			
		forward-word M-C-f			
		kill-word M-C-d			
		transpose-words M-C-t			
		history-expand-line M-^			

## Self-Insert

Hitting C v or <insert> runs quoted-insert \$key which inserts the escape sequence for \$key into the terminal. Running quoted-insert on these keys in sequence enters the following into the terminal. Successive C v keybinds seem to be handled recursively.

	000	001	010	011	100	101	110	111
	S	M	M S	C	C S	C M	C M S	
<insert>	^[[2-		^[[2;3-	^[[2;4-	^[[2;5-	^[[2;6-	^[[2;7-	^[[2;8-
C v	^v		^v					

The terminal application will intercept C v before it's given to the window manager (something like this), so you need to use C s v to paste.

In xkb, the shift modkey causes alphanumeric keybinds to be interpreted as their raw character codes. C s v literally inserts C v and AltGr ; into \$. Thus, I removed self-insert & do-lowercase-version binds from this reference.

## Escape Sequences

F1	^OP	F5	^[[15-	F9	^[[20-	home	^[[H	del	^[[3-
F2	^[[00	F6	^[[17-	F10	^[[21-	end	^[[F	bksp	^?
F3	^[[0R	F7	^[[18-	F11	^[[23-	pgup	^[[5-	bktab	^[[2
F4	^[[05	F8	^[[19-	F12	^[[24-	pgdn	^[[6-	esc	^[[

## Quality of Life Keybinds

abort C-g	digit-argument M [0-9-]	print-last-kbd-macro
capitalize-word M-c	downcase-word M-l	quoted-insert C-v, <insert>
character-search C ] \$char	dynamic-complete-history M c i	undo C _
backward M-C-] \$char	edit-and-execute-command C xC e	undo C xC u
	exchange-point-and-mark C xC x	upcase-word M-u
		yank
		nth-arg M C y
		pop M y

## Keybind Combos

### To avoid the strange unix-word-rubout

Keybinds involving C-w require setting:

```
bind ''\C-@:set-mark'
bind ''\e :set-mark'
bind ''\C-w:kill-region'
bind ''\ew:copy-region-as-kill'
```

### From middle of file or url path

Jump to last space, cut region, leaving the remainder of the path.

Then C-a varname=value; C-x C-x <right>

### From midline C-@ C-] | C-] | C-x C-x C-w

Set mark with C-@, search forward the next two | chars, then exchange-point-and-mark with C-x C-x.

This highlights the region. Then C-w to cut the text with kill-region.

### From midline =C-k C-a C-y

Swap order of commands

### From midline C-k C-a C-] | C-] | ... C-y

cut to end of line, search through line for | char and insert the tail of a bash pipeline. Simply repeating C-j doesn't work.

### From midline C-a export VAR=value;

Prefix command with environment var

### From midline C-k # C-y

Kill to end of line, insert # and paste. C-y to yank. Commented command content is still in history.

### Running print-last-keyboard-macro ...

Shows you what's recorded. You can make composite functions.

## Unmapped by default

alias-expand-line	dump	kill	previous
arrow-key-prefix	functions	region	screen-line
	macros	whole-line	
	variables		
backward	magic-space		
byte			
bash-vi-complete	emacs-editing-mode	menu	shell
	export-completions	complete	backward-kill-word
	fetch-history		
copy	backward-word	skip-csi-sequence	
	forward-word	tab-insert	
	region-as-kill	tty-status	
		universal-argument	
dabbrev-expand	forward		
	backward-delete-char		
	byte		
history	screen-line		
non-incremental			
delete	forward-search-history-again	unix	
	reverse-search-history-again	filename-rubout	
char-or-list	old-menu-complete		
	search-backward		
	search-forward		
	substring		
	search-backward		
	search-forward		

## Vi Bindings

For vi bindings ... you're on your own.

vi	vi	vi	vi
append	char-search	forward	redo
└ eol	column	└ bigword	replace
└ mode	complete	└ word	rubout
arg-digit	delete-to	└ fword	search-again
back-to-indent	edit-and-execute-command	└ fWord	set-mark
backward	editing-mode	└ goto-mark	subst
└ bigword	end	└ insert-beg	tilde-expand
└ word	└ bigword	└ insertion-mode	undo
bword	└ word	└ match	unix-word-rubout
bWord	eof-maybe	└ movement-mode	yank
change	eword	└ next-word	└ arg
└ case	└ eWord	└ overstrike-delete	└ pop
└ char	└ fetch-history	└ prev-word	└ to
└ to	└ first-print	└ put	