

Bash Keyboard Shortcuts Cheatsheet

By Dejan Panovski • Updated on Oct 24, 2026

Bash keyboard shortcuts for editing commands, searching history, completing words, and controlling foreground jobs.

Edit long commands and find earlier ones without retyping them. These shortcuts use Bash’s default Emacs editing mode. Hold Ctrl or Alt with the named key; if Alt does not work, press and release Esc, then press the other key. Sequences containing ‘then’ are pressed in order.

Move the Cursor

Move within the command you are editing, including commands that wrap across screen lines.

Ctrl+A	Go to the start of the line
Ctrl+E	Go to the end of the line
Ctrl+B	Move back one character
Ctrl+F	Move forward one character
Alt+B	Go to the start of this or the previous word
Alt+F	Go to the end of this or the next word

Readline treats letters and digits as words for Alt+B and Alt+F ; punctuation separates words.

Cut and Paste Text

Cut text goes into Readline’s kill ring, separate from your desktop clipboard.

Ctrl+U	Cut from the cursor back to the start of the line
Ctrl+K	Cut from the cursor to the end of the line
Ctrl+W	Cut the previous whitespace-delimited word
Alt+D	Cut to the end of this or the next word
Alt+Backspace	Cut back to the start of this or the previous word
Ctrl+Y	Paste the most recently cut text at the cursor
Alt+Y	After Ctrl+Y, cycle through earlier cuts

To cut the whole line, press Ctrl+A , then Ctrl+K . Ctrl+U alone leaves text after the cursor in place.

Correct and Undo Edits

These shortcuts change the current input line, not commands that have already run.

Backspace	Delete the character before the cursor
Ctrl+D	Delete the character at the cursor; see the empty-line caveat below
Ctrl+_	Undo the last edit; repeat to undo more
Ctrl+X then Ctrl+U	Another binding for undo
Alt+R	Undo all edits to this line
Ctrl+T	Swap the character before the cursor with the one at it; at line end, swap the last two
Alt+T	Swap the word before the cursor with the one after it; at line end, swap the last two
Alt+U, Alt+L, Alt+C	Uppercase, lowercase, or capitalize from the cursor to the end of the word
Ctrl+X then Ctrl+E	Open the line in your text editor; Bash runs it when you exit, so delete the text to cancel

Recall Commands and Arguments

Recall a command, edit it, and press Enter when you are ready to run it.

Ctrl+P or Up	Recall the previous command
Ctrl+N or Down	Recall the next command
Alt+<	Go to the oldest entry in the current history list
Alt+>	Return to the input line after the newest history entry
Alt+.	Insert the previous command’s last argument; repeat to use earlier commands

Alt+< and Alt+> use the literal < and > characters, which may require Shift on your keyboard.

Search Command History

Press `Ctrl+R` , then type part of a command to search backward through Bash history .

<code>Ctrl+R</code>	Start a reverse search or find an older match
<code>Ctrl+S</code>	Search forward, if terminal flow control is disabled
<code>Esc</code>	Leave the search with the match ready to edit
<code>Ctrl+G</code>	Cancel the search and restore the original input
<code>Enter</code>	Accept and run the matching command

Use `Esc` to review a match before executing it.

Complete Words

Bash completes command names, paths, variables, and other words according to the current context.

<code>Tab</code>	Complete the word before the cursor
<code>Tab twice</code>	List alternatives when completion is ambiguous, with default settings
<code>Alt+?</code>	List possible completions directly
<code>Alt+/<code></code></code>	Complete a filename
<code>Alt+\$</code>	Complete a shell variable name

Completion of command-specific options, such as Git subcommands, depends on the completion functions installed on your system.

Control the Terminal and Jobs

Signals act on the foreground job. A program can handle or ignore some signals, so interrupting does not always terminate it.

<code>Ctrl+C</code>	Send SIGINT to the foreground job; at the Bash prompt, cancel the input line
<code>Ctrl+Z</code>	Suspend the foreground job with SIGTSTP
<code>bg</code> then <code>Enter</code>	Resume the suspended job in the background
<code>fg</code> then <code>Enter</code>	Bring the current job to the foreground
<code>Ctrl+L</code>	Clear the screen and redraw the current input line
<code>Ctrl+D</code> on empty input	Send EOF; at a Bash prompt, normally exit the shell

`Ctrl+Z` suspends rather than ends a job. Use the Job Control cheatsheet for job numbers, `disown` , and logout behavior.

Check Your Bindings

Run these commands in an interactive Bash shell. Custom bindings can override the defaults above.

<code>set -o emacs</code>	Enable these Emacs-style bindings for this shell
<code>set -o vi</code>	Switch to vi editing mode, which uses different bindings
<code>bind -P</code>	List Readline functions and their current key bindings
<code>bind -v</code>	Show Readline variable settings, including editing mode
<code>Ctrl+X</code> then <code>Ctrl+R</code>	Reload the Readline initialization file
<code>stty -a</code>	Show terminal settings, including signal keys and flow control

Readline reads the file named by `INPUTRC` , or `~/.inputrc` when that variable is unset. If that file cannot be read, it falls back to `/etc/inputrc` , so start a new `~/.inputrc` with `$include /etc/inputrc` to keep your distribution’s bindings.

Terminal Caveats

The terminal can intercept a key before Bash receives it.

<code>Alt</code> shortcuts do not work	Use <code>Esc</code> , then the other key; on macOS, you can also configure Option to send Meta
<code>Ctrl+S</code> pauses output	<code>Ctrl+Q</code> resumes it when software flow control is enabled
You want forward history search	Run <code>stty -ixon</code> to disable software flow control for this terminal; <code>stty ixon</code> enables it again
<code>Ctrl+D</code> exits unexpectedly	It deletes only when input is present; an empty line sends EOF. <code>set -o ignoreeof</code> prevents a single EOF from exiting Bash
Paste behaves differently	<code>Ctrl+Y</code> uses the kill ring; terminal clipboard paste is often <code>Ctrl+Shift+V</code> on Linux or <code>Cmd+V</code> on macOS

Related Guides

Use these references for scripting syntax, history settings, and the full list of Readline commands.

Bash Cheatsheet	Variables, conditionals, loops, and shell scripting
history Command in Linux	History files, expansion, filtering, and settings
GNU Bash Readline Commands	Official shortcut definitions and function names
GNU Bash Readline Configuration	Input files and custom key bindings