

[bash](#), [howto](#), [commandlineediting](#), [keyboard](#), [shortcut](#)



## Bash - Command Line Editing

### Sources

- <https://ss64.com/bash/syntax-keyboard.html>
- <https://www.howtogeek.com/howto/ubuntu/keyboard-shortcuts-for-bash-command-shell-for-ubuntu-debian-suse-redhat-linux-etc/>

### Basic Shortcuts

You can use the following key combinations to edit and recall commands:

Shortcut	Command
CTRL+L	Clear the screen.
CTRL+W	Delete the word starting at cursor.
CTRL+U	Clear the line i.e. Delete the all words from command line.
↑, ↓	Recall commands (see command history).
TAB	Auto-complete files, directory, command names and much more.
CTRL+R	Search through previously used commands (see command history)
CTRL+C	Cancel currently running commands.
CTRL+T	Swap the last two characters before the cursor.
ESC+T	Swap the last two words before the cursor.

### Bash Keyboard Shortcuts

#### Moving the cursor

Use the following shortcuts to quickly move the cursor around the current line while typing a command.

Shortcut	Command
Ctrl+A, POS1	Go to the beginning of the line (Home)
Ctrl+E, END	Go to the End of the line (End)
ALT+B	Back (left) one word
ALT+F	Forward (right) one word
Ctrl+F	Forward one character
Ctrl+B	Backward one character
Ctrl+Xx	(double tap 'X' ) Move between the beginning of the line and the current position of the cursor. This allows you to press Ctrl+Xx to return to the start of the line, change something, and then press Ctrl+Xx to go back to your original cursor position. To use this shortcut, hold the Ctrl and tap X twice.

#### Editing

Shortcut	Command
CTRL+L	Clear the Screen, similar to the clear command
ALT+BACKSPACE	Delete the Word before the cursor.
CTRL+D	Delete character under the cursor
ALT+D	Delete from actual cursor position to the end of word
CTRL+H	Delete character before the cursor (Backspace)
CTRL+W	Cut the Word before the cursor to the clipboard.
CTRL+K	Cut the Line after the cursor to the clipboard.
CTRL+U	Cut/delete the Line before the cursor to the clipboard.
ALT+T	Swap current word with previous
CTRL+T	Swap the last two characters before the cursor (typo).
ESC+T	Swap the last two words before the cursor.
CTRL+Y	Paste the last thing to be cut (yank)
ALT+U	UPPER capitalize every character from the cursor to the end of the current word.

Shortcut	Command
<code>ALT+L</code>	Lower the case of every character from the cursor to the end of the current word.
<code>ALT+C</code>	Capitalize the character under the cursor and move to the end of the word.
<code>ALT+R</code>	Cancel the changes and put back the line as it was in the history (revert).
<code>CTRL+Z</code>	Undo
<code>TAB</code>	Tab completion for file/directory names For example, to move to a directory 'sample1'; Type <code>cd sam</code> ; then press <code>TAB</code> and <code>ENTER</code> . type just enough characters to uniquely identify the directory you wish to open.

## Special keys: Tab, Backspace, Enter, Esc

Text Terminals send characters (bytes), not key strokes. Special keys such as **Tab**, **Backspace**, **Enter** and **Esc** are encoded as control characters. Control characters are not printable, they display in the terminal as `^` and are intended to have an effect on applications.

Keys	Control Character
<code>Ctrl+I</code>	<code>TAB</code>
<code>Ctrl+J</code>	Newline
<code>Ctrl+M</code>	<code>ENTER</code>
<code>Ctrl+[</code>	<code>ESC</code>

Many terminals will also send control characters for keys in the digit row:

Shortcut	Result
<code>CTRL+2</code>	<code>^@</code>
<code>CTRL+3</code>	<code>^[ ESCAPE</code>
<code>CTRL+4</code>	<code>^\</code>
<code>CTRL+5</code>	<code>^]</code>
<code>CTRL+6</code>	<code>^^</code>
<code>CTRL+7</code>	<code>^_ UNDO</code>
<code>CTRL+8</code>	<code>^? Backward-delete-char</code>

`CTRL+V` tells the terminal to not interpret the following character, so `CTRL+V CTRL+I` will display a **TAB** character, similarly `CTRL+V ENTER` will display the escape sequence for the Enter key: `^M`.

## History

Shortcut	Command
<code>CTRL+R</code> or <code>↑</code>	Recall the last command including the specified character(s). Searches the command history as you type. Equivalent to : <code>vim ~/.bash_history</code> .
<code>CTRL+P</code> or <code>↓</code>	Previous command in history (i.e. walk back through the command history)
<code>CTRL+N</code>	Next command in history (i.e. walk forward through the command history)
<code>CTRL+Shift</code>	Go back to the next most recent command. (beware to not execute it from a terminal because this will also launch its XOFF).
<code>CTRL+O</code>	Execute the command found via <code>Ctrl+R</code> or <code>Ctrl+S</code>
<code>CTRL+G</code>	Escape from history searching mode
<code>!!</code>	Repeat last command
<code>!abc</code>	Run last command starting with abc
<code>!abc:p</code>	Print last command starting with abc
<code>!\$</code>	Last argument of previous command
<code>ALT+.</code>	Last argument of previous command
<code>!*</code>	All arguments of previous command
<code>^abc^def</code>	Run previous command, replacing abc with def

## Process control

Shortcut	Command
<code>CTRL+C</code>	Interrupt (kill) the current foreground process running in in the terminal. This sends the <b>SIGINT signal</b> to the process, which is technically just a request—most processes will honor it, but some may ignore it.
<code>CTRL+D</code>	Close the bash shell. This sends an <b>EOF (End-of-file)</b> marker to bash, and bash exits when it receives this marker. This is similar to running the exit command.
<code>CTRL+Z</code>	Suspend the current foreground process running in bash. This sends the <b>SIGTSTP signal</b> to the process. To return the process to the foreground later, use the <b>fg process_name</b> command.

## Controlling the screen

The following shortcuts allow you to control what appears on the screen.

Shortcut	Command
CTRL+L	Clear the screen. This is similar to running the “clear” command.
CTRL+S	Stop all output to the screen. This is particularly useful when running commands with a lot of long, verbose output, but you don't want to stop the command itself with
CTRL+Q	Resume output to the screen after stopping it with <code>Ctrl+Shift</code> .

## Emacs mode vs Vi Mode

All the above assume that bash is running in the **default Emacs setting**, if you prefer this can be switched to **Vi shortcuts** instead.

Set **Vi Mode** in bash:

```
set -o vi
```

Set **Emacs Mode** in bash:

```
set -o emacs
```

“...emacs, which might be thought of as a thermonuclear word processor” ~ [Emacs vs. Vi Wiki](#)

## Related

- [fg](#) - Bring a command to the foreground.
- [vi editor](#) - [A one page reference to the vi editor](#).
- `~/.bash_history` - Text file with command history.
- [Terminals Are Weird](#) - How and why of terminal keybindings.
- Equivalent [Windows Keyboard shortcuts](#)

From:  
<https://wiki.nanoscopic.de/> - **nanoscopic wiki**

Permanent link:  
<https://wiki.nanoscopic.de/doku.php/pages/howtos/bash/bash-command-line-editing?rev=1613047591>

Last update: **2021/02/11 12:46**

