|  |
| --- |
|  **Zsh Cheat Sheet** |
| **Installing Zsh Commands** |  | **Control Flow and Script Execution** |
| *sudo dnf* Installs Zsh with user’s package manager on*install zsh* Fedora, CentOs and RHEL | *break* Exits from a loop or switch statement |

*sudo apt install zsh*

Installs Zsh with user’s package manager on

Debian and Ubuntu

*bye* Exits the shell

*sudo port*

*install zsh*

Installs Zsh with user’s package manager on

MacOS and user can install it using MacPorts

*continue* Skips the remaining iterations of a loop and starts the next iteration

*brew install*

*zsh*

Installs Zsh with user’s package manager on

MacOS and user can install it using

*return* Exits a function with an optional return value

**Navigation Commands**

[*exit*](https://linuxsimply.com/exit-command-in-linux/)Exits the shell with an optional exit status

[*cd*](https://linuxsimply.com/cd-command-in-linux/)Changes the current directory

*cd ..* Moves up one level in the directory hierarchy

*cd /* Changes the current directory to the root directory

*cd ~* Changes the current directory to your home directory

*cd -P* Changes the current directory and resolves symbolic links

*cd -L* Changes the current directory without resolving symbolic links

*fc* Manages and edits command history

*shift*

Shifts positional parameters to the left by a

specified number

*test* Evaluates conditional expressions and performs tests

*bg [job ...]* Resumes suspended jobs in the background

**Process Control Commands**

Sets or displays actions to be taken when a

signal is received

*trap*

*cd*

*/path/to/dir ectory*

Changes the current directory to the specified directory path

*disown [job ...]*

Removes jobs from the job table, allowing them to continue running

*code .* Opens the current directory in Visual Studio Code

*dirs* Displays or manipulates the directory stack

*fg [job ...]* Brings jobs to the foreground

[*jobs*](https://linuxsimply.com/jobs-command-in-linux/)Lists active jobs and their statuses

*explorer .*

Opens the current directory in the default

file explorer (on Windows)

[*kill*](https://linuxsimply.com/kill-command-in-linux/)

[*[options]*](https://linuxsimply.com/kill-command-in-linux/)Sends signals to processes or job IDs

[*job ...*](https://linuxsimply.com/kill-command-in-linux/)

[*ls*](https://linuxsimply.com/ls-command-in-linux/)Lists files and directories in the current

directory

*suspend [-*

*f]*

Suspends the current shell session

*open .*

Opens the current directory in the default

file explorer (on macOS)

*pushd* Changes the current directory and pushes

Displays a line of text or the value of a variable

on the terminal

[*echo*](https://linuxsimply.com/echo-command-in-linux/)

**Input/Output Commands**

*wait [job* Waits for background jobs to complete

*...]*

the current directory onto the directory

*popd*

Removes entries from the directory stack

and changes the current directory to the popped directory

[*pwd*](https://linuxsimply.com/pwd-command-in-linux/)Shows the current working directory *print*

Displays text or variables on the terminal, and

various formatting options and can be used to control the output appearance

[*export*](https://linuxsimply.com/export-command-in-linux/)Sets environment variables

Displays or sets shell variables

[*declare*](https://linuxsimply.com/declare-command-in-linux/)

**Variable & Environment**

*readonly* Marks variables as read-only *read*

Reads input from the user or from a file and assigns it to variables and allows user to prompt for input, store the entered values, and perform further processing based on the

*typeset*

Declares variables with specific attributes

*unset* Unsets variables or functions

*unsetopt*

Disables shell options

**Command Execution and**

**Environment**

Executes a block of code based on a condition,

If the condition is true, the code within the "if"

*if*

**Conditional Execution and**

Controls terminal settings and behavior, and -

*ttyctl -fu* fu options force a flush of the terminal output buffer, ensuring that all pending output is

displayed immediately

*noglob*

Executes a command without performing globbing on arguments

*then*

Specifies the start of the code block to be executed if the condition of an "if" statement is true

*rehash* Updates the internal hash table of executable commands

[*source*](https://linuxsimply.com/source-command-in-linux/)Executes commands from a file in the current shell session

[*time*](https://linuxsimply.com/time-command-in-linux/)Displays the system and user times for the current shell session

**Command Information and Location**

*unhash* Removes commands from the internal hash table

*fi* Marks the end of an "if" statement

*else*

Specifies the code block to be executed if the

condition of the preceding "if" statement is false

*do*

Specifies the start of the code block to be

executed in a "for" or "while" loop

*done* Marks the end of a "for" or "while" loop

*for*

Executes a block of code repeatedly for a

specified number of iterations or for each item in a list

*until* Executes a block of code repeatedly until a certain condition is true

*where*

Displays all locations where a command is

defined

[*which*](https://linuxsimply.com/which-command-in-linux/)Displays the path to the executable file of a command

Views, edits, re-executes, or changes the

order of previously entered commands

*fc*

**History and Command Editing**

Executes a block of code repeatedly as long as

a certain condition is true

*while*

*getln*

*whence* Displays information about a command,

including its type and location

**Shell Built-In Commands**

[*alias*](https://linuxsimply.com/alias-command-in-linux/)

Defines or lists aliases for commands

Displays the command history, showing a list of previously executed commands along with their line numbers

*getopts* Allows to process command-line arguments

*optstring* and options, and enables to create more

interactive and flexible shell scripts

[*unalias*](https://linuxsimply.com/unalias-command-in-linux/)

*builtin*

Removes aliases for commands

Executes a built-in shell command

*disable* Disables shell built-in commands or functions. [*history*](https://linuxsimply.com/history-command-in-linux/)

Displays the command history, showing a list of previously executed commands along with their line numbers

*false* Returns a non-zero exit status *functions* Lists or defines functions *unfunction* Removes defined shell functions

[*enable*](https://linuxsimply.com/enable-command-in-linux/)

Enables shell built-in commands or functions

*hash*

Displays or modifies command hash table

*set* Sets or displays shell options or positional parameters

*setopt* Sets or displays shell options

*file* Displays information about the specified file(s)

Creates copies of files and directories

[*cp*](https://linuxsimply.com/cp-command-in-linux/)

**File and Directory Management**

[*mkdir*](https://linuxsimply.com/mkdir-command-in-linux/)Creates a new directory

*mkcd* Creates a new directory and changes the current directory to it in a single command

[*mv*](https://linuxsimply.com/mv-command-in-linux/)Moves or renames files and directories

[*rm*](https://linuxsimply.com/rm-command-in-linux/)Removes a file

*true*

Returns a zero exit status

*rm -r*

Removes a directory and its contents

recursively

*type* Displays information about a command

[*touch*](https://linuxsimply.com/touch-command-in-linux/)Creates an empty file or updates the access and modification timestamps of an existing

*typeset*

Declares or displays shell variables with

additional attributes

*zmv*

Allows users to rename multiple files using

complex patterns and expressions.

*unset* Unsets shell variables or functions

*zmvn*

Allows users to files rename using natural sorting order, which handles numerical sequences in a more human-friendly way

*zcalc* Performs arithmetic calculations directly in the shell

*unsetopt*

Unsets shell options

**Miscellaneous**

*zmodload -* Loads one or more shared modules into the

*dL* Zsh shell

**Loading & Unloading Modules**

*zmodload -* Loads one or more modules and generates

*e* error messages if loading fails

*xargs*

Allows users to apply another command to a

list of arguments, similar to xargs but with enhanced features

*zmodload*

*[ -a [ -bcp* Loads modules, specifying various loading

*[ -I ] ] ] [ -iL* options

*] …*

*zprof*

*zstyle*

Provides profiling information for Zsh startup files, allowing user to optimize their loading time

Customizes the behavior and appearance of the shell by setting various options and styles

*zmodload*

*-u [ -*

*abcdp [ -I ]*

*zmodload -* Unloads a specific module or all loaded

*d [ -L ] [*

*name ]*

modules

Unloads modules, specifying various unloading options

*zargs* Applies another command to a list of arguments

Profiles Zsh startup files, helps to optimize

*zmodload - d name*

Unloads a module and its dependencies

*zprof*

*zstyle*

their loading time by identifying potential performance bottlenecks

Configures and customizes various aspects of Zsh's behavior, including shell options, command completion, and highlighting

*zmodload -* Lists all available built-in modules

*zmodload -* Unloads a module and its dependencies and

*ud name [*

*dep ... ]*

unloads any unused modules

*ab [ -L ]*

*vared*

Provides a convenient way to edit variables in the shell using the editor specified by the EDITOR environment variable

*zmodload -*

*ab [ -i ] name [*

*builtin ... ]*

*zmodload -*

*ub [ -i ]* Unloads one or more built-in modules

*builtin ...*

Loads one or more built-in modules and generates error messages if loading fails

*push-line*

Pushes the current command line onto the

buffer and allows you to continue editing it, useful for complex command constructions

*autoload* Allows users to lazily load Zsh functions

Allow you to create new instances of the

*zmodload -*

*ac [ -IL ]*

*zmodload -*

Lists all available conditional modules

*clone/detach*

*zcompile*

current shell, either as a forked child process (clone) or as a detached background process (detach)

Compiles Zsh scripts into a more efficient format, which can improve their execution time

*ac [ -iI ] name [ cond ... ] zmodload - uc [ -iI ]*

*cond ...*

*zmodload -* Lists all available parameter expansion

*ap [ -L ]* modules

Loads one or more conditional modules and generates error messages if loading fails

Unloads one or more conditional modules

*zpty* Provides a way to create and interact with pseudo-terminals from within Zsh

Allows users to create TCP and UDP

*zmodload - ap [ -i ]*

Loads one or more parameter expansion

*ztcp*

connections directly from the shell. It can be used for various network-related tasks

*name [ parameter*

modules and generates error messages if loading fails

*zsocket*

Allows users to create and interact with Unix

domain sockets from within Zsh. This can be useful for inter-process communication

*... ]*

*zmodload -*

*up [ -i ]* Unloads one or more parameter expansion

*parameter* modules

*...*

*zparseopts*

*zregexparse*

*zsh-mime- setup*

Simplifies the handling of command-line

options and arguments in Zsh scripts by providing a convenient way to parse and process them

Provides a way to parse text using regular

expressions and extract specific elements of interest. It is particularly useful for text Configures default programs and actions

associated with different MIME types in the Zsh environment

*zmodload - a [ -L ]*

*zmodload - a [ -i ]*

*name [ zmodload - ua [ -i ]*

*builtin ...*

*zmodload -*

*e [ string ...* Evaluates the given strings as Zsh code

*]*

Lists all available modules

Loads one or more modules and generates error messages if loading fails

Unloads one or more modules