HOSTINGER

Three. Two. Online

Bash Cheat Sheet

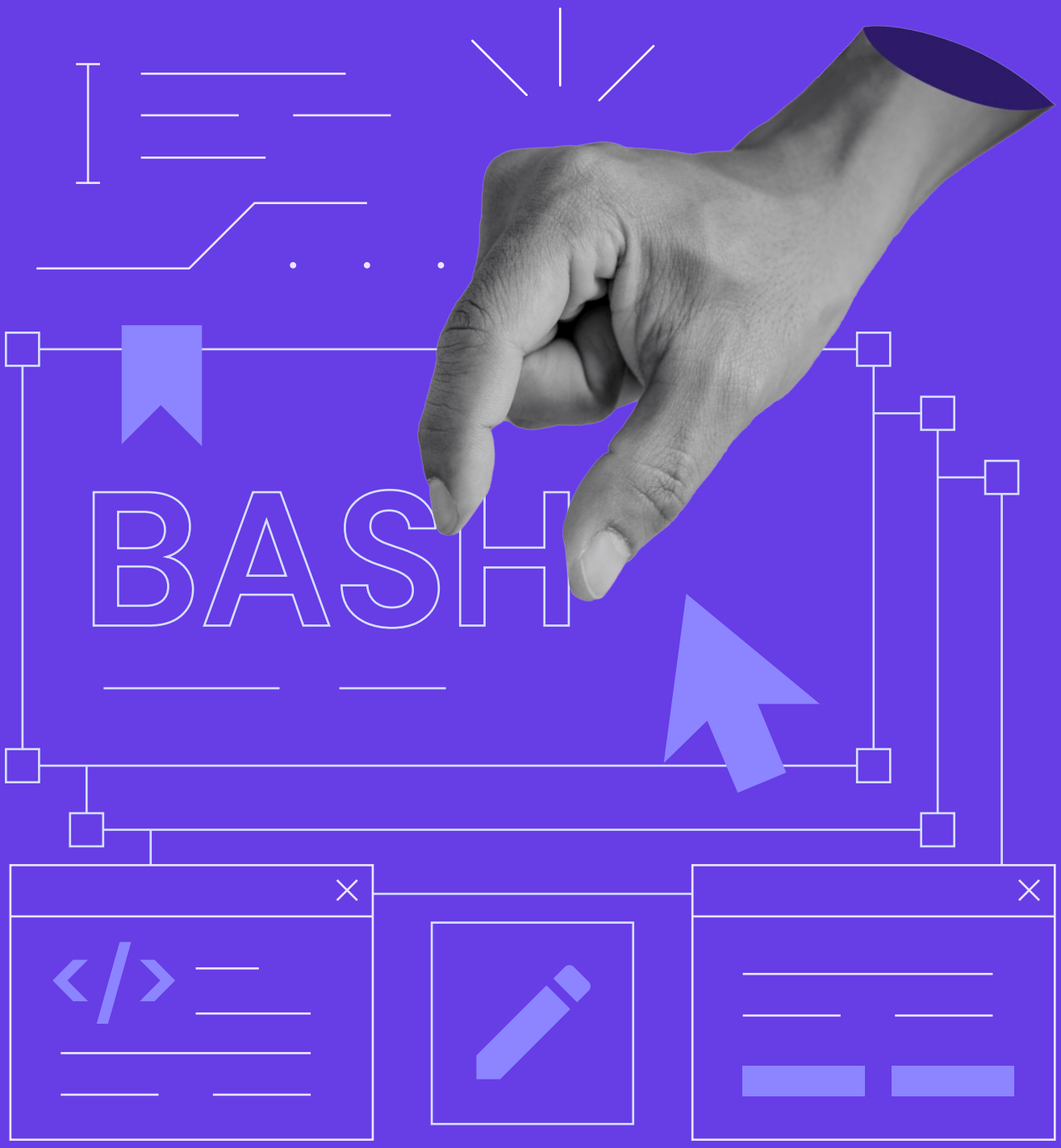

Table of Content

Click or tap to jump to any section:

1. File Management
   - Creation
   - Moving
   - Reading
2. Deletion
   - Compression and Decompression
   - Replacing in Files
3. Search in Files
   - Directories
   - Navigation
4. Creation
   - Moving
   - Deletion
5. Symbolic Links
   - Permissions
   - Table
   - Popular File Permission Examples
6. Popular Directory Permission Examples
   - Permission Management
7. Arrays
   - Creation
   - Adding Elements
   - Printing Out
   - Deletion
8. Resource Usage and Processes
   - Shutdown and Reboot
9. Scheduled Tasks
   - Crontab Syntax
   - Possible Values
   - Possible Symbols
10. Tmux Terminal Multiplexer
11. HTTP Requests
    - Network and DNS
12. Secure Shell Protocol (SSH)

Bash Cheat Sheet

File Management

Creation

`mktemp`

Creates a temporary file with a random name. Guarantees that the new file doesn't exist

`touch new_file.txt`

Creates a new file

`touch new_file.txt new_file_2.txt`

Creates both **new_file.txt** and **new_file_2.txt**

`touch {new_file, new_file_2}.txt`

Creates both **new_file.txt** and **new_file_2.txt**

`touch file_name_{a..c}`

Creates **file_name_a, file_name_b**, and **file_name_c**

`touch file_name_{1..3}`

Creates **file_name_1, file_name_2**, and **file_name_3**

Moving

`cp file_name.txt file_name_copy.txt`

Creates a copy of **file_name.txt** named **file_name_copy.txt** in the same working directory

`mv file_name.txt new_file_copy.txt`

Moves **file_name.txt** to **new_file_copy.txt** in the same directory, renaming the file

Reading

`head file_name.txt`

Prints first 10 lines of a file

`tail file_name.txt`

Prints last 10 lines of a file

`cat file_name.txt`

Prints full contents of a file

```
less file_name.txt
```

Prints a part of file contents

```
wc file_name.txt
```

Prints number of lines words and characters in the file

Deletion

```
rm file_name.txt
```

Removes a file

```
rm -f file_name.txt
```

Removes a file ignoring non-existent files

Compression and Decompression

```
zip archive_name.zip file_name_1.php file_name_2.php
```

```
tar -cvf archive_name.tar file_name_1.php file_name_2.php
```

```
tar -zcf archive_name.tar.gz file_name_1.php file_name_2.php
```

Archives separate files (depending on the format)

```
zip -r archive_name.zip directory
```

```
tar -cvf archive_name.tar directory
```

```
tar -zcf archive_name.tar.gz directory
```

Archives a whole directory (depending on the format)

```
unzip archive_name.zip
```

```
tar -xvf archive_name.tar
```

```
tar -zxvf archive_name.tar.gz
```

Extracts an archive (depending on the format)

Replacing in Files

```
sed -i 's/search_query/replace_query/' file_name.txt
```

Modifies the content of the original file if the replacement query exists in the file

```
sed 's/search_query/replace_query/g' file_name.txt > new_file_name.txt
```

Replaces **search_query** with **replace_query** in **file_name.txt** and saves everything in **new_file_name.txt**

```
sed 's/search_query/replace_query/g' file_name.txt
```

Replaces **search_query** with **replace_query** in **file_name.txt**

Search in Files

`grep 'contents' /file_name.txt`

Searches for **contents** inside **file_name.txt**

`grep 'contents' /directory -r`

Searches for **contents** inside **directory** recursively

`grep 'contents_1\|contents_2' /directory -R`

Searches for **contents** or **contents_2** inside **directory** recursively

`grep 'contents' /directory -v`

Displays only the lines that don't match **contents**

`grep 'contents' /directory -i`

Performs case-insensitive search

`grep 'contents' /directory -i`

Performs case-insensitive search

`grep 'contents' /directory -x`

Matches the entire line and prints it out

`grep 'contents' /directory -n`

Displays line numbers along with the results

`grep 'contents' /directory -l`

Only displays files that match **contents** query

`grep 'contents' /directory -L`

Only shows files that don't match **contents** query

Directories

Navigation

`cd directory`

Goes to the listed sub-directory

`cd` / `cd ~`

Goes to the home directory

`cd ..`

Goes the directory one level up from the current directory

`cd -`

Goes to the previous directory

`ls`

Lists all directories

`ls -a`

Lists all directories, including hidden ones

`ls -l`

Shows where symbolic links are pointing

`ls -l -h`

Lists all directories in long format, **-h** flag uses the human-readable format

`ls -t`

List directories by modification time, showing newest first

`tree`

Shows directory and file trees

`tree -a`

Shows directory and file trees, including hidden ones

`tree -d`

Shows directory tree

`stat file.txt`

Lists file's size alongside created and modified timestamps

`stat directory`

Lists directory's size alongside created and modified timestamps

`pwd`

Shows current directory path

Creation

`mkdir new_directory`

Creates a new directory

`mkdir new_directory_1 directory_2`

Creates multiple new directories

`mkdir -p parent/child/nested_child`

Creates nested directories

`mkdir -p {dir_one, dir_two}/nested`

Creates multiple nested directories

`mktemp -d`

Creates a temporary directory with a random name. Guarantees that the new directory doesn't exist

Moving

`mv old_directory new_directory`

Moves **old_directory** to **new_directory**

`cp directory directory_copy`

Copies **directory** to **directory_copy**

`cp -r directory directory_copy`

Copies **directory** to **directory_copy** recursively

Deletion

`rmdir directory`

Removes **directory**

`rm -r directory`

Removes **dicrectory** recursively

Symbolic Links

`ln -s path link`

Creates a symbolic link called **link** to the **path** directory

`ln -s -f path link`

Overwrites existing symbolic link called **link**

Permissions

Table

Octal	Decimal	Permission	Symbolic representation
000	0	No permissions	- - -
001	1 (0 + 0 + 1)	Execute permission	- - x
010	2 (0 + 2 + 0)	Write permission	- w -
011	3 (0 + 2 + 1)	Write and execute permissions	- w x
100	4 (4 + 0 + 0)	Read permission	r - -
101	5 (4 + 0 + 1)	Read and execute permissions	r - x
110	6 (4 + 2 + 0)	Read and write permissions	r w -
111	7 (4 + 2 + 1)	Read, write and execute permissions	r w x

Popular File Permission Examples

Value	Explanation
777	All possible permissions. Not recommended for security reasons.
755	File owner can read, write, and execute. Other users can read and execute.
700	File owner can read, write, and execute the file. Other users have no permissions.
666	All users can read and write the file.
644	File owner can read and write. Other users can only read the file.
600	File owner can read and write the file. Other users have no permissions.

Popular Directory Permission Examples

Value	Explanation
777	All possible permissions. Not recommended for security reasons.
755	Directory owner has full permissions. Other users can list the directory but won't be able to manage the files.
700	Directory owner has full permissions. Other users have no permissions.

Permission Management

`ls -l`

Displays directory contents in a long format, which displays both permissions and ownership

`chown :new_group file_name`

Changes group for **file_name**

`chmod u=rwx, g=r, o-rwx file_name`

Sets **file_name** read, write, execute permissions for the owner and other users, leaving read-only permissions for the group

`chmod u+x file_name.txt`

Sets the user permissions to execute

`chmod u+x, g+x, o+x file_name.txt`

`chmod a+x file_name.txt`

`chmod +x file_name.txt`

Sets everyone's permissions to execute

`chown username:new_group file_name`

Changes both the owner and group for **file_name**

`chown username file_name`

Changes the owner for **file_name**

`chmod 777 -R directory`

Sets read, write and execute permissions for everyone in **directory** recursively

`chmod g+x file_name.txt`

Sets the group permissions to execute

Arrays

Creation

```
indexed_array = (element_1, element_2, element_3)
```

Creates an index array

```
declare -A associative_array = ([key_1] = element_1, [key_2] = element_2, [key_3] = element_3)
```

Creates an associative array

Adding Elements

```
indexed_array += (new_element)
```

Adds a new element to an indexed array

```
associative_array += ([key_1] = new_element
```

Adds a new element to an associative array (note that the key needs to be provided as well)

Printing Out

```
echo ${indexed_array[0]}
```

Prints out the first array element

```
echo ${indexed_array[@]}
```

Prints out the whole array

```
echo ${!associative_array[@]}
```

Prints out all the keys for an associative array

Deletion

```
unset indexed_array[2]
```

Removes the third element from an indexed array

```
unset associative_array[key]
```

Removes the key element from an associative array

Resource Usage and Processes

`top` / `htop`

Lists out all the processes interactively

`ps all`

Lists out all currently running processes

`pidof process_name`

Prints out the process ID of **process_name**

`nice -n 10 process_name`

Changes the priority to 10 for **process_name**

`renice 10 2468`

Changes the priority to 10 for a process that has process ID **2468**

`ps -o ni 2468`

Displays the priority for a process ID **2468**

`kill 2468`

Kills a process that has process ID **2468**

`killall process_name`

Kills all processes that have **process_name** in their name

`jobs`

Shows all background processes

`jobs -p`

Shows all processes jobs alongside their IDs

`lsof`

Shows all open files and processes that use them

`free`

Displays memory usage

`du`

Shows current directory, sub-directories and file sizes

`du /directory/sub-directory`

Lists specified directory, sub-directories and file sizes

`df`

Shows disks alongside their used and available space

Shutdown and Reboot

`shutdown`

Shuts the system down after one minute

`shutdown now`

Immediately shuts down the system

`shutdown +10`

Shuts the system down after 10 minutes

`shutdown -r`

`reboot`

Immediately reboots the system

`shutdown -r +10`

Reboots the system after 10 minutes

`shutdown -c`

Cancels a shutdown or reboot

`reboot -f`

Forces a reboot

Scheduled Tasks

Crontab Syntax

*	*	*	*	*	Command
Minute	Hour	Day of month	Month	Day of week	

Possible Values

Field	Possible values
Minute	0-59
Hour	0-23
Day of month	1-31
Month	1-12
Day of week	0-6. 0 depicts Sunday. In some systems, a value of 7 represents Sunday instead
Command	Command to execute

Possible Symbols

Symbol	Meaning	Example
* (asterisk)	Select all possible values in a field	Place * in the **hour** field to run the task every hour
– (hyphen)	A comma is used to separate multiple values	0,3,5 in the **day of week** field will make the task run on Sunday and Wednesday
/ (separator)	Used to set a range of values	10-15 in the **day of month** field will run the task from the 10th to the 15th day of the month
L	Used in the **day of month** or **day of week** fields	*/10 in the **hour** field will make the task run every 10 hours
W	W is used to determine the closest weekday	1L in the **day of week** field will run the task on the last Monday of a given month

Symbol	Meaning	Example
# (hash)	Used to determine the **day of week**	2#3 in the **day of month** field will make the task run on the third Tuesday of the month
? (question mark)	Used in the **day of month** and **day of week** fields	? in the **day of month** field will read as **no specific value**

`crontab -e`

Used to edit system crontabs. This command will make a new crontab if it has not been created yet

`crontab -l`

Used to view crontab entries (cron jobs), and display system crontab file contents

`crontab -r`

Will remove the current crontab file

`crontab -i`

Will show a prompt before removing a user's crontab

`* * * * * cat /home/hello_world.sh`

Schedules a job to run every minute

`@daily cat /home/hello_world.sh`

Schedules a background job to run every day

`00 08-17 * * * cat /home/hello_world.sh`

Schedules a job to run every weekday, including weekends, from 8am to 5pm

`@reboot cat /home/hello_world.sh`

Schedules a job to run after each system reboot

`@monthly cat /home/hello_world.sh`

Schedules a job to run at the beginning of each month

`0 12,15,17,19,21 * * * cat /home/hello_world.sh`

Schedules a job to be run five times a day at 12pm, 3pm, 5pm, 7pm and, 9pm

Tmux Terminal Multiplexer

`tmux`

Start tmux

`<prefix> + %`

Split panes into two horizontally

`<prefix> + "`

Split panes into two vertically

<prefix> + <arrow keys>

Navigate between panes

<prefix> + D

Close panes

<prefix> + C

Create a new tmux window

<prefix> + N

Go to the next window

<prefix> + P

Navigate to the previous window

HTTP Requests

curl https://domain.tld

Returns response body for domain.tld

curl -i https://domain.tld

Returns response body for domain.tld and includes status code and HTTP headers

curl -o file.txt https://domain.tld

Outputs to a text file

curl -H|--header "User-Agent: Agent" https://domain.tld

Adds an HTTP header

Network and DNS

ip addr

Shows all IP addresses on a system

ip route show

Shows all IP addresses to router

ping domain.tld

Sends multiple ICMP protocol ping requests

ping -c 15 -i 3 domain.tld

Pings the domain 10 times, 3 seconds apart

netstat -l

Shows all open ports

netstat -i

Shows all open ports with in/out usage

traceroute domain.tld

Displays all servers the network traffic goes through

nmap 0.0.0.0

Scans for the 1,000 most commonly open ports on localhost

nmap 255.255.255.255

Scans for the 1,000 most commonly open ports on remote IP address

nmap 0.0.0.0 -p1-65535

Scans for open ports on localhost between 1 and 65535

host example.net

Display IPv4 and IPv6 addresses for domain.tld

dig example.net

Display complete DNS information

`dig example.net +short`

Display complete DNS in short format

`dig example.net txt`

Query TXT records

`dig example.net cname`

Query CNAME records

`dig example.net ns`

Query NS records

`dig example.net A`

Query A records

`dig example.net MX`

Query MX records

Secure Shell Protocol (SSH)

`ssh hostname`

Connects to hostname using current username via default SSH port 22

`ssh root@255.255.255.255`

Connects via given username and IP via default SSH port 22

`ssh root@255.255.255.255 -p 1023`

Connects via given username and IP via given SSH port