**bashoneliners.com**

Welcome to bashoneliners.com, a growing collection of practical and well-explained Bash tips and tricks. We are committed to review every contributed one-liner to ensure its high quality, practical usefulness, and a solid explanation of how it works. We want to document one-liners for frequent (non-trivial) tasks executed in the shell, written in the best possible way, following good practices known in the industry. To get the latest Bash one-liners, follow [@bashoneliners](https://twitter.com/bashoneliners) on Twitter. If you find any problems, [please report on GitHub Issues](https://github.com/janosgyerik/bashoneliners/issues).

* [Tags](https://www.bashoneliners.com/oneliners/tags/)
* [Popular](https://www.bashoneliners.com/oneliners/popular/)
* [Active](https://www.bashoneliners.com/oneliners/active/)
* [Newest](https://www.bashoneliners.com/oneliners/newest/)

|  |  |
| --- | --- |
| 0 score | [**Get a wide info of all pods on all namespaces with Kubectl**](https://www.bashoneliners.com/oneliners/315/)   $ kubectl get pods -A -o wide  Sept. 15, 2022, 6:42 p.m. — [adawolfs](https://www.bashoneliners.com/oneliners/users/457/) |
| 0 score | [**Oneliner checksum validation**](https://www.bashoneliners.com/oneliners/314/)   $ sha512sum filename.iso | grep -oE ".\*\ " | (read checksum; [ "$Checksum" == checksumCheck ] && printf "\033[0;32mValid" || printf "\033[0;31mInvalid")  Aug. 24, 2022, 10:43 a.m. — [Hussain-Aqeel](https://www.bashoneliners.com/oneliners/users/453/) | |

|  |  |
| --- | --- |
| 0 score | [**Move file and cd with one command**](https://www.bashoneliners.com/oneliners/313/)   $ mv file dir && cd "$\_" && pwd  April 16, 2022, 9:55 a.m. — [Bryan-netizen](https://www.bashoneliners.com/oneliners/users/442/) |
| 0 score | [**Report disk usage by file type**](https://www.bashoneliners.com/oneliners/312/)   $ find . -type f -empty -prune -o -type f -printf "%s\t" -exec file --brief --mime-type '{}' \; | awk 'BEGIN {printf("%12s\t%12s\n","bytes","type")} {type=$2; a[type]+=$1} END {for (i in a) printf("%12u\t%12s\n", a[i], i)|"sort -nr"}'  Dec. 28, 2020, 10:09 p.m. — [joeashcraft](https://www.bashoneliners.com/oneliners/users/403/) | |

|  |  |
| --- | --- |
| 0 score | [**Find with invert match - e.g. find every file that is not mp3**](https://www.bashoneliners.com/oneliners/311/)   $ find . -name '\*' -type f -not -path '\*.mp3'  Nov. 23, 2020, 10:21 p.m. — [sandreas](https://www.bashoneliners.com/oneliners/users/424/) |
| -1 score | [**Convert pip list --outdated for reuse in pip install -U**](https://www.bashoneliners.com/oneliners/310/)   $ pip install -U $(pip list --outdated 2> /dev/null | grep -v 'Version' | grep -v '------' | awk '{printf $1 " " }' && echo)  April 16, 2020, 1:04 p.m. — [docdyhr](https://www.bashoneliners.com/oneliners/users/405/) | |

|  |  |  |
| --- | --- | --- |
| 0 score | [**Convert pip list --outdated for reuse in pip install -U**](https://www.bashoneliners.com/oneliners/309/)   $ python3 -m pip install -U $(python3 -m pip list outdated 2> /dev/null | grep -v 'Version' | grep -v '\-\-\-\-\-\-' | awk '{printf $1 " " }' && echo)  April 16, 2020, 10:55 a.m. — [docdyhr](https://www.bashoneliners.com/oneliners/users/405/) | |
| 0 score | [**Recursively remove all "node\_modules" folders**](https://www.bashoneliners.com/oneliners/308/)   $ find . -name "node\_modules" -exec rm -rf '{}' +  March 29, 2020, 8:50 a.m. — [tg-z](https://www.bashoneliners.com/oneliners/users/401/) |

|  |  |
| --- | --- |
| 0 score | [**Read other user's shell history, and convert epoch to human-readable date**](https://www.bashoneliners.com/oneliners/306/)   $ cat /home/john/.bash\_history | awk '/#[0-9]\*$/ {split($0, arr, "#"); print "#", strftime("%c",arr[2]);getline; print; }'  March 10, 2020, 6 p.m. — [joeashcraft](https://www.bashoneliners.com/oneliners/users/403/) |
| 1 score | [**Outputs list of $PATH dirs sorted by line length**](https://www.bashoneliners.com/oneliners/305/)   $ echo -e ${PATH//:/\\n} | awk '{print length, $0}' | sort -n | cut -f2- -d' '  March 7, 2020, 7:39 a.m. — [tg-z](https://www.bashoneliners.com/oneliners/users/401/) |

|  |  |
| --- | --- |
| 0 score | [**Generate a sequence of numbers**](https://www.bashoneliners.com/oneliners/304/)   $ seq 1 10  Jan. 17, 2020, 10:09 p.m. — [penkoad](https://www.bashoneliners.com/oneliners/users/398/) |
| 0 score | [**Prints "Good" in green if the sha256 sum of a file matches the sum listed in a dist/checksum file and "Bad" in red if it does not match**](https://www.bashoneliners.com/oneliners/303/)   $ [[ $(sha256sum \_downloaded\_file\_ | cut -d' ' -f 1) == $(grep -Po '\b[a-zA-Z0-9]{64}\b' \_checksum\_fle\_) ]] && printf "\033[0;32mGood" || printf "\033[0;31mBad"  Dec. 13, 2019, 8:10 p.m. — [ratfink417](https://www.bashoneliners.com/oneliners/users/393/) | |

|  |  |  |
| --- | --- | --- |
| 0 score | [**Listen to a song from youtube**](https://www.bashoneliners.com/oneliners/302/)   $ listen-to-yt() { if [[ -z "$1" ]]; then echo "Enter a search string!"; else mpv "$(youtube-dl --default-search 'ytsearch1:' \"$1\" --get-url | tail -1)"; fi }  Dec. 8, 2019, 6:46 p.m. — [MKamnikar](https://www.bashoneliners.com/oneliners/users/366/) | |
| 0 score | [**(Pseudo)random 6-digit number**](https://www.bashoneliners.com/oneliners/301/)   $ let N=0 N=N+10\*\*{0..5}\*${RANDOM:0:1}  Dec. 7, 2019, 3:26 p.m. — [CK](https://www.bashoneliners.com/oneliners/users/389/) |

|  |  |  |
| --- | --- | --- |
| 0 score | [**Compute factorial of positive integer using only built-ins**](https://www.bashoneliners.com/oneliners/300/)   $ bang() ( eval let N=1 N\*={1..$1} ; echo $N; )  Nov. 29, 2019, 9:06 p.m. — [CK](https://www.bashoneliners.com/oneliners/users/389/) | |
| 2 score | [**Generate a sequence of numbers**](https://www.bashoneliners.com/oneliners/299/)   $ printf '%s\n' {1..10}  Nov. 25, 2019, 8:11 a.m. — [CK](https://www.bashoneliners.com/oneliners/users/389/) |

|  |  |  |
| --- | --- | --- |
| 0 score | [**Recursively compute factorial of positive integer using only built-ins**](https://www.bashoneliners.com/oneliners/298/)   $ bang() ( IFS=\\*; let N=$1-1 k="$\*" && bang $N $k || echo ${2-1} )  Nov. 25, 2019, 8:06 a.m. — [CK](https://www.bashoneliners.com/oneliners/users/389/) | |
| 1 score | [**Using a single sudo to run multiple && arguments**](https://www.bashoneliners.com/oneliners/297/)   $ sudo -s <<< 'apt update -y && apt upgrade -y'  Oct. 27, 2019, 7:02 p.m. — [secure-debian](https://www.bashoneliners.com/oneliners/users/340/) |

|  |  |  |
| --- | --- | --- |
| 1 score | | [**Generate a sequence of numbers**](https://www.bashoneliners.com/oneliners/296/)   $ for i in {1..10};do echo $i;done  Oct. 25, 2019, 12:48 a.m. — [diego](https://www.bashoneliners.com/oneliners/users/387/) |
| 0 score | [**List docker log sizes and remind how to empty them**](https://www.bashoneliners.com/oneliners/295/)   $ docker ps -qa | xargs docker inspect --format='{{.LogPath}}' | xargs sudo du -hl; sleep 1; printf "\r\n echo '' > \$(docker inspect --format={{.LogPath}} container\_name\_or\_id) \r\n\*\*\*\*\*COPY\*\*\*\*ABOVE\*\*\*\*TO\*\*\*\*\*\*CLEAR\*\*\*\*\*LOG\*\*\*\*\*CHANGE\*\*\*CONTAINER\_ID\*\*\*\*\*\*\* \r\n \n"  June 18, 2019, 8:06 p.m. — [lockntross](https://www.bashoneliners.com/oneliners/users/370/) | | |

|  |  |
| --- | --- |
| 2 score | [**Find files/dirs modified within a given period**](https://www.bashoneliners.com/oneliners/292/)   $ find . -type d -newermt "2019-01-01" \! -newermt "2019-02-01" -exec ls -ld {} \;  May 20, 2019, 6:48 p.m. — [fenchu](https://www.bashoneliners.com/oneliners/users/363/) |
| 1 score | [**Scan all open ports without any required program**](https://www.bashoneliners.com/oneliners/291/)   $ for i in {1..65535}; do (echo < /dev/tcp/127.0.0.1/$i) &>/dev/null && printf "\n[+] Open Port at\n: \t%d\n" "$i" || printf "."; done  May 8, 2019, 4:56 p.m. — [Goeks1](https://www.bashoneliners.com/oneliners/users/332/) |

|  |  |
| --- | --- |
| 0 score | [**Extract audio only from video files using ffmpeg**](https://www.bashoneliners.com/oneliners/290/)   $ ffmpeg -i video.any -vn -acodec libvorbis audio.ogg  April 21, 2019, 10:55 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |
| 2 score | [**Generate a random 32 characters password**](https://www.bashoneliners.com/oneliners/286/)   $ tr -dc 'a-zA-Z0-9~!@#$%^&\*\_()+}{?></";.,[]=-' < /dev/urandom | fold -w 32 | head -n 1  March 24, 2019, 6:39 p.m. — [cheuv25](https://www.bashoneliners.com/oneliners/users/355/) | |

|  |  |
| --- | --- |
| 1 score | [**Using tcpdump with port ranges and file count/size**](https://www.bashoneliners.com/oneliners/279/)   $ sudo /usr/sbin/tcpdump -i any -s 0 -n -Z <user\_name> -C 500 -W 100 -w /home/<user\_name>/$(hostname).pcap -f '(port (# or # or # or # or # or # or ...) or portrange <start>-<end>)' &>/dev/null |

**bashoneliners.com**

Welcome to bashoneliners.com, a growing collection of practical and well-explained Bash tips and tricks. We are committed to review every contributed one-liner to ensure its high quality, practical usefulness, and a solid explanation of how it works. We want to document one-liners for frequent (non-trivial) tasks executed in the shell, written in the best possible way, following good practices known in the industry. To get the latest Bash one-liners, follow [@bashoneliners](https://twitter.com/bashoneliners) on Twitter. If you find any problems, [please report on GitHub Issues](https://github.com/janosgyerik/bashoneliners/issues).

* [Tags](https://www.bashoneliners.com/oneliners/tags/)
* [Popular](https://www.bashoneliners.com/oneliners/popular/)
* [Active](https://www.bashoneliners.com/oneliners/active/)
* [Newest](https://www.bashoneliners.com/oneliners/newest/)

|  |  |
| --- | --- |
| 1 score | [**Compare (diff) vim-generated backup of a file with that file's current version.**](https://www.bashoneliners.com/oneliners/278/)   $ diff~() { diff "$1"~ "$1"; }  March 12, 2019, 2:52 a.m. — [juanolopezgerena](https://www.bashoneliners.com/oneliners/users/352/) |
| 1 score | [**Count the total number of hours of your music collection**](https://www.bashoneliners.com/oneliners/277/)   $ find . -print0 | xargs -0 -P 40 -n 1 sh -c 'ffmpeg -i "$1" 2>&1 | grep "Duration:" | cut -d " " -f 4 | sed "s/.$//" | tr "." ":"' - | awk -F ':' '{ sum1+=$1; sum2+=$2; sum3+=$3; sum4+=$4; if (sum4 > 100) { sum3+=1; sum4=0 }; if (sum3 > 60) { sum2+=1; sum3=0 }; if (sum2 > 60) { sum1+=1; sum2=0 } if (NR % 100 == 0) { printf "%.0f:%.0f:%.0f.%.0f\n", sum1, sum2, sum3, sum4 } } END { printf "%.0f:%.0f:%.0f.%.0f\n", sum1, sum2, sum3, sum4 }'  March 1, 2019, 9:09 p.m. — [pingiun](https://www.bashoneliners.com/oneliners/users/351/) |

|  |  |  |
| --- | --- | --- |
| 0 score | [**Print the list of your Git commits this month**](https://www.bashoneliners.com/oneliners/276/)   $ git log --since='last month' --author="$(git config user.name)" --oneline  Feb. 25, 2019, 6:57 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) | |
| 0 score | [**Store the output of find in an array**](https://www.bashoneliners.com/oneliners/275/)   $ mapfile -d $'\0' arr < <(find /path/to -print0)  Feb. 18, 2019, 7:48 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |

|  |  |
| --- | --- |
| 1 score | [**Clear the Exim Queue (remove all messages)**](https://www.bashoneliners.com/oneliners/274/)   $ exim -bp | exiqgrep -i | xargs exim -Mrm  Feb. 11, 2019, 8:58 a.m. — [Robert](https://www.bashoneliners.com/oneliners/users/347/) |
| 1 score | [**Check if a text snippet is valid C code**](https://www.bashoneliners.com/oneliners/273/)   $ gcc -fsyntax-only -xc - <<< "text snippet"  Feb. 10, 2019, 8:12 a.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) | |

|  |  |
| --- | --- |
| 1 score | [**Inspect the HTTP headers of a website**](https://www.bashoneliners.com/oneliners/272/)   $ curl -I amazon.com  Feb. 8, 2019, 10:56 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |
| 2 score | [**Search man pages and present a PDF**](https://www.bashoneliners.com/oneliners/271/)   $ man -k . | awk '{ print $1 " " $2 }' | dmenu -i -p man | awk '{ print $2 " " $1 }' | tr -d '()' | xargs man -t | ps2pdf - - | zathura -  Dec. 18, 2018, 11:31 a.m. — [Jab2870](https://www.bashoneliners.com/oneliners/users/258/) | |

|  |  |
| --- | --- |
| 4 score | [**Find all log files modified 24 hours ago, and zip them**](https://www.bashoneliners.com/oneliners/270/)   $ find . -type f -mtime +1 -name "\*.log" -exec zip -m {}.zip {} \; >/dev/null  Nov. 9, 2018, 10:04 a.m. — [TrongTan124](https://www.bashoneliners.com/oneliners/users/337/) |
| 4 score | [**List IP addresses connected to your server on port 80**](https://www.bashoneliners.com/oneliners/269/)   $ netstat -tn 2>/dev/null | grep :80 | awk '{print $5}' | cut -d: -f1 | sort | uniq -c | sort -nr | head  Sept. 26, 2018, 11:10 p.m. — [Goeks1](https://www.bashoneliners.com/oneliners/users/332/) |

|  |  |  |
| --- | --- | --- |
| 1 score | [**Random 6-digit number**](https://www.bashoneliners.com/oneliners/268/)   $ python -c 'import random; print(random.randint(0,1000000-1))'  Sept. 19, 2018, 10:42 p.m. — [johntellsall](https://www.bashoneliners.com/oneliners/users/329/) | |
| 1 score | [**Very fast history search with Ripgrep**](https://www.bashoneliners.com/oneliners/267/)   $ rh() { rg "$1" ~/.bash\_history }  Sept. 18, 2018, 7 p.m. — [johntellsall](https://www.bashoneliners.com/oneliners/users/329/) |

|  |  |  |
| --- | --- | --- |
| 0 score | [**While loop to pretty print system load (1, 5 & 15 minutes)**](https://www.bashoneliners.com/oneliners/266/)   $ while :; do date; awk '{printf "1 minute load: %.2f\n", $1; printf "5 minute load: %.2f\n", $2; printf "15 minute load: %.2f\n", $3}' /proc/loadavg; sleep 3; done  Sept. 5, 2018, 8:41 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) | |
| 1 score | [**Scan entire Git repo for dangerous Amazon Web Service IDs**](https://www.bashoneliners.com/oneliners/265/)   $ git grep -Ew '[A-Z0-9]{20}'  Sept. 5, 2018, 8:30 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |

|  |  |
| --- | --- |
| 0 score | [**Scan entire Git repos for dangerous Amazon Web Service IDs**](https://www.bashoneliners.com/oneliners/264/)   $ git ls-tree --full-tree -r --name-only HEAD | xargs egrep -w '[A-Z0-9]{20}'  Aug. 31, 2018, 10:29 p.m. — [johntellsall](https://www.bashoneliners.com/oneliners/users/329/) |
| 0 score | [**While loop to pretty print system load (1, 5 & 15 minutes)**](https://www.bashoneliners.com/oneliners/263/)   $ while [ 1 == 1 ]; do cat /proc/loadavg | awk '{printf "1 minute load: %.2f\n", $(NF-5)}' && cat /proc/loadavg |awk '{printf "5 minute load: %.2f\n", $(NF-3)}' && cat /proc/loadavg |awk '{printf "15 minute load: %.2f\n", $(NF-2)}'; sleep 3; date; done  Aug. 30, 2018, 8:54 a.m. — [peek2much3](https://www.bashoneliners.com/oneliners/users/330/) |

|  |  |
| --- | --- |
| 1 score | [**Dump all AWS IAM users/roles to a Terraform file for editing / reusing in another environment**](https://www.bashoneliners.com/oneliners/262/)   $ echo iamg iamgm iamgp iamip iamp iampa iamr iamrp iamu iamup | AWS\_PROFILE=myprofile xargs -n1 terraforming  Aug. 28, 2018, 12:38 a.m. — [johntellsall](https://www.bashoneliners.com/oneliners/users/329/) |
| 0 score | [**Organise image by portrait and landscape**](https://www.bashoneliners.com/oneliners/261/)   $ mkdir "portraits"; mkdir "landscapes"; for f in ./\*.jpg; do WIDTH=$(identify -format "%w" "$f")> /dev/null; HEIGHT=$(identify -format "%h" "$f")> /dev/null; if [[ "$HEIGHT" > "$WIDTH" ]]; then mv "$f" portraits/ ; else mv "$f" landscapes/ ; fi; done  Aug. 23, 2018, 2:09 p.m. — [Jab2870](https://www.bashoneliners.com/oneliners/users/258/) |

|  |  |
| --- | --- |
| 0 score | [**Create a txt files with 10000 rows**](https://www.bashoneliners.com/oneliners/260/)   $ for FILE in \*.full ; do split -l 100000 $FILE; mv -f xaa `echo "$FILE" | cut -d'.' -f1`.txt; rm -f x\*; done  Aug. 22, 2018, 2:02 p.m. — [Kifli88](https://www.bashoneliners.com/oneliners/users/328/) |
| 1 score | [**List open processes ordered by it's number of open files**](https://www.bashoneliners.com/oneliners/259/)   $ ps -ef |awk '{ print $2 }' \ |tail -n +2 \ |while read pid; do echo "$pid $(lsof -p $pid |wc -l)"; done \ |sort -r -n -k 2 \ |while read pid count; do echo "$pid $count $(ps -o command= -p $pid)"; done  Aug. 22, 2018, 1:21 p.m. — [cddr](https://www.bashoneliners.com/oneliners/users/327/) |

|  |  |
| --- | --- |
| 0 score | [**Remove all container from an specific network (docker)**](https://www.bashoneliners.com/oneliners/258/)   $ docker ps -a -f network=$NETWORK --format='{{.ID}}' | xargs docker rm -f  Aug. 17, 2018, 4:38 p.m. — [gatero](https://www.bashoneliners.com/oneliners/users/325/) |
| 0 score | [**Up all docker services as detached mode over all immediate subdirectories**](https://www.bashoneliners.com/oneliners/257/)   $ for dir in $(ls -d \*/); do eval $(cd $PWD/$dir && docker-compose up -d && cd ..); done;  Aug. 17, 2018, 4:31 p.m. — [gatero](https://www.bashoneliners.com/oneliners/users/325/) |

|  |  |  |
| --- | --- | --- |
| 0 score | | [**Find and replace string inside specific files**](https://www.bashoneliners.com/oneliners/256/)   $ grep -ril '$SEARCH\_PATTERN' src | sed -i 's/$FIND\_PATTERN/$REPLACE\_PATTERN/g'  Aug. 17, 2018, 4:18 p.m. — [gatero](https://www.bashoneliners.com/oneliners/users/325/) |
| 0 score | [**Puppet/Bash: test compare json objects.**](https://www.bashoneliners.com/oneliners/255/)   $ unless => "client\_remote=\"$(curl localhost:9200/\_cluster/settings | python -c \"import json,sys;obj=json.load(sys.stdin);print(obj['persistent']['search']['remote'])\")\"; new\_remote=\"$( echo $persistent\_json | python -c \"import json,sys;obj=json.load(sys.stdin);print(obj['persistent']['search']['remote'])\")\"; [ \"$client\_remote\" = \"$new\_remote\" ]",  July 27, 2018, 8:37 p.m. — [cjedwa](https://www.bashoneliners.com/oneliners/users/321/) | |

|  |  |
| --- | --- |
| 0 score | [**Print wifi access points sorted by signal**](https://www.bashoneliners.com/oneliners/254/)   $ iw dev IFACE scan | egrep "SSID|signal" | awk -F ":" '{print $2}' | sed 'N;s/\n/:/' | sort  June 16, 2018, 5:37 a.m. — [kazatca](https://www.bashoneliners.com/oneliners/users/308/) |

Welcome to bashoneliners.com, a growing collection of practical and well-explained Bash tips and tricks. We are committed to review every contributed one-liner to ensure its high quality, practical usefulness, and a solid explanation of how it works. We want to document one-liners for frequent (non-trivial) tasks executed in the shell, written in the best possible way, following good practices known in the industry. To get the latest Bash one-liners, follow [@bashoneliners](https://twitter.com/bashoneliners) on Twitter. If you find any problems, [please report on GitHub Issues](https://github.com/janosgyerik/bashoneliners/issues).

* [Tags](https://www.bashoneliners.com/oneliners/tags/)
* [Popular](https://www.bashoneliners.com/oneliners/popular/)
* [Active](https://www.bashoneliners.com/oneliners/active/)
* [Newest](https://www.bashoneliners.com/oneliners/newest/)

|  |  |
| --- | --- |
| 2 score | [**Kill a process running on port 8080**](https://www.bashoneliners.com/oneliners/253/)   $ lsof -i :8080 | awk '{l=$2} END {print l}' | xargs kill  June 15, 2018, 4:18 a.m. — [jamestomasino](https://www.bashoneliners.com/oneliners/users/307/) |
| 0 score | [**Delete all untagged Docker images**](https://www.bashoneliners.com/oneliners/252/)   $ docker images -q -f dangling=true | xargs --no-run-if-empty --delim='\n' docker rmi  June 15, 2018, 1:12 a.m. — [penguincoder](https://www.bashoneliners.com/oneliners/users/305/) | |

|  |  |  |
| --- | --- | --- |
| 1 score | [**Take values from a list (file) and search them on another file**](https://www.bashoneliners.com/oneliners/251/)   $ for ITEM in $(cat values\_to\_search.txt); do (egrep $ITEM full\_values\_list.txt && echo $ITEM found) | grep "found" >> exit\_FOUND.txt; done  May 16, 2018, 3:20 p.m. — [ManuViorel](https://www.bashoneliners.com/oneliners/users/302/) | |
| -1 score | [**Delete all untagged Docker images**](https://www.bashoneliners.com/oneliners/250/)   $ docker rmi $(docker images -f "dangling=true" -q)  April 27, 2018, 2:50 p.m. — [stefanobaghino](https://www.bashoneliners.com/oneliners/users/288/) |

|  |  |
| --- | --- |
| 1 score | [**Have script run itself in a virtual terminal**](https://www.bashoneliners.com/oneliners/249/)   $ tty >/dev/null || { urxvt -e /bin/sh -c "tty >/tmp/proc$$; while test x; do sleep 1; done" & while test ! -f /tmp/proc$$; do sleep .1; done; FN=$(cat /tmp/proc$$); rm /tmp/proc$$; exec >$FN 2>$FN <$FN; }  March 9, 2018, 2:56 a.m. — [openiduser111](https://www.bashoneliners.com/oneliners/users/143/) |
| 2 score | [**Big CSV > batches > JSON array > CURL POST data with sleep**](https://www.bashoneliners.com/oneliners/248/)   $ cat post-list.csv | split -l 30 - --filter='jq -R . | jq --slurp -c .' | xargs -d "\n" -I % sh -c 'curl -H "Content-Type: application/json" -X POST -d '"'"'{"type":1,"entries":%}'"'"' http://127.0.0.1:8080/purge-something && sleep 30'  March 7, 2018, 12:12 p.m. — [pratham2003](https://www.bashoneliners.com/oneliners/users/291/) |

|  |  |
| --- | --- |
| 2 score | [**List all packages with at least a class defined in a JAR file**](https://www.bashoneliners.com/oneliners/247/)   $ jar tf "$1" | grep '/.\*\.class$' | xargs dirname | sort -u | tr / .  Feb. 19, 2018, 12:13 p.m. — [stefanobaghino](https://www.bashoneliners.com/oneliners/users/288/) |
| 2 score | [**Output an arbitrary number of open TCP or UDP ports in an arbitrary range**](https://www.bashoneliners.com/oneliners/246/)   $ comm -23 <(seq "$FROM" "$TO") <(ss -tan | awk '{print $4}' | cut -d':' -f2 | grep "[0-9]\{1,5\}" | sort | uniq) | shuf | head -n "$HOWMANY"  Feb. 9, 2018, 3:51 p.m. — [stefanobaghino](https://www.bashoneliners.com/oneliners/users/288/) | |

|  |  |  |
| --- | --- | --- |
| 0 score | [**Source without circular reference**](https://www.bashoneliners.com/oneliners/245/)   $ [ ! "${LIB}" ] && ( readonly LIB; . "${ $( cd $( dirname $0 ) && pwd ) }/<path\_to>/LIB.sh" )  Jan. 24, 2018, 4:30 p.m. — [dhsrocha](https://www.bashoneliners.com/oneliners/users/286/) | |
| 4 score | [**Ternary conditional clause**](https://www.bashoneliners.com/oneliners/244/)   $ [ test\_statement ] && ( then\_statement ) || ( else\_statement );  Jan. 22, 2018, 5:27 p.m. — [dhsrocha](https://www.bashoneliners.com/oneliners/users/286/) |

|  |  |
| --- | --- |
| 3 score | [**Get executed script's current working directory**](https://www.bashoneliners.com/oneliners/243/)   $ CWD=$(cd "$(dirname "$0")" && pwd)  Jan. 22, 2018, 4:55 p.m. — [dhsrocha](https://www.bashoneliners.com/oneliners/users/286/) |
| 3 score | [**Random Git Commit**](https://www.bashoneliners.com/oneliners/242/)   $ git commit -m "$(w3m whatthecommit.com | head -n 1)"  Jan. 5, 2018, 4:55 p.m. — [Jab2870](https://www.bashoneliners.com/oneliners/users/258/) | |

|  |  |  |
| --- | --- | --- |
| 1 score | [**Blackhole ru zone**](https://www.bashoneliners.com/oneliners/241/)   $ echo "address=/ru/0.0.0.0" | sudo tee /etc/NetworkManager/dnsmasq.d/dnsmasq-ru-blackhole.conf && sudo systemctl restart network-manager  Nov. 14, 2017, 2:12 p.m. — [olshek\_](https://www.bashoneliners.com/oneliners/users/283/) | |
| 0 score | [**Remove new lines from files and folders**](https://www.bashoneliners.com/oneliners/240/)   $ rename 's/[\r\n]//g' \*  Sept. 30, 2017, 10:07 p.m. — [moverperfect](https://www.bashoneliners.com/oneliners/users/280/) |

|  |  |
| --- | --- |
| 2 score | [**Retrieve dropped connections from firewalld journaling**](https://www.bashoneliners.com/oneliners/239/)   $ sudo journalctl -b | grep -o "PROTO=.\*" | sed -r 's/(PROTO|SPT|DPT|LEN)=//g' | awk '{print $1, $3}' | sort | uniq -c  Sept. 14, 2017, 5:10 p.m. — [FoxBuru](https://www.bashoneliners.com/oneliners/users/276/) |
| 1 score | [**Kill a process running on port 8080**](https://www.bashoneliners.com/oneliners/238/)   $ lsof -i :8080 | awk 'NR > 1 {print $2}' | xargs --no-run-if-empty kill  Sept. 1, 2017, 8:31 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |

|  |  |
| --- | --- |
| 1 score | [**Kill a process running on port 8080**](https://www.bashoneliners.com/oneliners/237/)   $ lsof -i :8080 | awk '{print $2}' | tail -n 1 | xargs kill  Aug. 18, 2017, 8:22 a.m. — [kimbethwel](https://www.bashoneliners.com/oneliners/users/272/) |
| 1 score | [**Get the latest Arch Linux news**](https://www.bashoneliners.com/oneliners/236/)   $ w3m https://www.archlinux.org/ | sed -n "/Latest News/,/Older News/p" | head -n -1  Aug. 15, 2017, 10:35 a.m. — [Jab2870](https://www.bashoneliners.com/oneliners/users/258/) | |

|  |  |  |
| --- | --- | --- |
| 2 score | | [**Make a new folder and cd into it.**](https://www.bashoneliners.com/oneliners/235/)   $ mkcd(){ NAME=$1; mkdir -p "$NAME"; cd "$NAME"; }  Aug. 3, 2017, 6:49 a.m. — [PrasannaNatarajan](https://www.bashoneliners.com/oneliners/users/270/) |
| 1 score | [**Listen to the radio (radio2 in example)**](https://www.bashoneliners.com/oneliners/234/)   $ mpv http://a.files.bbci.co.uk/media/live/manifesto/audio/simulcast/hls/uk/sbr\_med/llnw/bbc\_radio\_two.m3u8  July 19, 2017, 2:44 p.m. — [Jab2870](https://www.bashoneliners.com/oneliners/users/258/) | | |

|  |  |  |
| --- | --- | --- |
| 1 score | [**Go up to a particular folder**](https://www.bashoneliners.com/oneliners/233/)   $ alias ph='cd ${PWD%/public\_html\*}/public\_html'  July 18, 2017, 6:07 p.m. — [Jab2870](https://www.bashoneliners.com/oneliners/users/258/) | |
| 3 score | [**Open another terminal at current location**](https://www.bashoneliners.com/oneliners/232/)   $ $TERMINAL & disown  July 18, 2017, 3:04 p.m. — [Jab2870](https://www.bashoneliners.com/oneliners/users/258/) |

|  |  |  |
| --- | --- | --- |
| 1 score | [**Preserve your fingers from cd ..; cd ..; cd..; cd..;**](https://www.bashoneliners.com/oneliners/231/)   $ up(){ DEEP=$1; for i in $(seq 1 ${DEEP:-"1"}); do cd ../; done; }  June 28, 2017, 5:40 p.m. — [alireza6677](https://www.bashoneliners.com/oneliners/users/262/) | |
| 0 score | [**Get the HTTP status code of a URL**](https://www.bashoneliners.com/oneliners/230/)   $ curl -Lw '%{http\_code}' -s -o /dev/null -I SOME\_URL  June 19, 2017, 11:15 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |

|  |  |
| --- | --- |
| 0 score | [**Corporate random bullshit generator (cbsg)**](https://www.bashoneliners.com/oneliners/229/)   $ curl -s http://cbsg.sourceforge.net/cgi-bin/live | grep -Eo '^<li>.\*</li>' | sed s,\</\\?li\>,,g | shuf -n 1 | cowsay  June 7, 2017, 4:11 p.m. — [Jab2870](https://www.bashoneliners.com/oneliners/users/258/) |

* [Tags](https://www.bashoneliners.com/oneliners/tags/)
* [Popular](https://www.bashoneliners.com/oneliners/popular/)
* [Active](https://www.bashoneliners.com/oneliners/active/)
* [Newest](https://www.bashoneliners.com/oneliners/newest/)

|  |  |
| --- | --- |
| 0 score | [**Generate a sequence of numbers**](https://www.bashoneliners.com/oneliners/228/)   $ perl -e 'print "$\_\n" for (1..10);'  May 30, 2017, 2:47 p.m. — [abhinickz6](https://www.bashoneliners.com/oneliners/users/255/) |
| 1 score | [**List the content of a GitHub repository without cloning it**](https://www.bashoneliners.com/oneliners/227/)   $ svn ls https://github.com/user/repo/trunk/some/path  May 21, 2017, 6:01 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) | |

|  |  |
| --- | --- |
| 1 score | [**Delete static and dynamic arp for /24 subnet**](https://www.bashoneliners.com/oneliners/226/)   $ for i in {1..254}; do arp -d 192.168.0.$i; done  Oct. 21, 2016, 5:07 a.m. — [dennyhalim.com](https://www.bashoneliners.com/oneliners/users/246/) |
| 8 score | [**List status of all GIT repos**](https://www.bashoneliners.com/oneliners/225/)   $ find ~ -name ".git" 2> /dev/null | sed 's/\/.git/\//g' | awk '{print "-------------------------\n\033[1;32mGit Repo:\033[0m " $1; system("git --git-dir="$1".git --work-tree="$1" status")}'  Oct. 16, 2016, 11:19 p.m. — [uMt](https://www.bashoneliners.com/oneliners/users/245/) | |

|  |  |
| --- | --- |
| 1 score | [**Shuffle lines**](https://www.bashoneliners.com/oneliners/224/)   $ ... | perl -MList::Util=shuffle -e 'print shuffle <>;'  Jan. 31, 2016, 9:02 p.m. — [openiduser81](https://www.bashoneliners.com/oneliners/users/107/) |
| 0 score | [**Create an array of CPU frequencies in GHz**](https://www.bashoneliners.com/oneliners/223/)   $ cpus=($({ echo scale=2; awk '/cpu MHz/ {print $4 " / 1000"}' /proc/cpuinfo; } | bc))  Dec. 28, 2015, 9:02 p.m. — [openiduser146](https://www.bashoneliners.com/oneliners/users/146/) | |

|  |  |
| --- | --- |
| 0 score | [**Test git archive before actually creating an archive // fake dry run**](https://www.bashoneliners.com/oneliners/222/)   $ git archive master some/project/subdir | tar t  Dec. 22, 2015, 2:29 p.m. — [openiduser146](https://www.bashoneliners.com/oneliners/users/146/) |
| 2 score | [**Convert all flac files in dir to mp3 320kbps using ffmpeg**](https://www.bashoneliners.com/oneliners/221/)   $ for FILE in \*.flac; do ffmpeg -i "$FILE" -b:a 320k "${FILE[@]/%flac/mp3}"; done;  Sept. 20, 2015, 5:45 p.m. — [Orkan](https://www.bashoneliners.com/oneliners/users/233/) | |

|  |  |
| --- | --- |
| 1 score | [**Preserve your fingers from cd ..; cd ..; cd..; cd..;**](https://www.bashoneliners.com/oneliners/220/)   $ upup(){ DEEP=$1; [ -z "${DEEP}" ] && { DEEP=1; }; for i in $(seq 1 ${DEEP}); do cd ../; done; }  June 9, 2015, 3:09 p.m. — [andreaganduglia](https://www.bashoneliners.com/oneliners/users/229/) |
| 1 score | [**Get number of all Python Behave scenarios (including all examples from Scenario Outlines)**](https://www.bashoneliners.com/oneliners/218/)   $ behave -d | grep "scenarios passed" | cut -d, -f4 | sed -e 's/^[[:space:]]\*//' | sed 's/untested/scenarios/g'  April 17, 2015, 2:21 p.m. — [openiduser188](https://www.bashoneliners.com/oneliners/users/225/) |

|  |  |  |
| --- | --- | --- |
| 4 score | [**Ban all IPs that attempted to access phpmyadmin on your site**](https://www.bashoneliners.com/oneliners/217/)   $ grep "phpmyadmin" $path\_to\_access.log | grep -Po "^\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}" | sort | uniq | xargs -I% sudo iptables -A INPUT -s % -j DROP  April 2, 2015, 8:58 a.m. — [openiduser187](https://www.bashoneliners.com/oneliners/users/224/) | |
| -1 score | [**Get a free shell account on a community server**](https://www.bashoneliners.com/oneliners/216/)   $ sh <(curl hashbang.sh | gpg)  March 15, 2015, 9:49 a.m. — [lrvick](https://www.bashoneliners.com/oneliners/users/222/) |

|  |  |  |
| --- | --- | --- |
| 0 score | [**Shuffle lines**](https://www.bashoneliners.com/oneliners/215/)   $ seq 5 | shuf  March 12, 2015, 7:58 a.m. — [openiduser184](https://www.bashoneliners.com/oneliners/users/221/) | |
| 8 score | [**Generate a sequence of numbers**](https://www.bashoneliners.com/oneliners/214/)   $ echo {01..10}  March 1, 2015, 12:04 a.m. — [Elkku](https://www.bashoneliners.com/oneliners/users/218/) |

|  |  |
| --- | --- |
| 4 score | [**Run a command and copy its output to clipboard (Mac OSX)**](https://www.bashoneliners.com/oneliners/213/)   $ echo "Here comes the output of my failing code" | tee >(pbcopy)  Feb. 28, 2015, 11:53 p.m. — [Elkku](https://www.bashoneliners.com/oneliners/users/218/) |
| 5 score | [**Nmap scan every interface that is assigned an IP**](https://www.bashoneliners.com/oneliners/212/)   $ ifconfig -a | grep -Po '\b(?!255)(?:\d{1,3}\.){3}(?!255)\d{1,3}\b' | xargs nmap -A -p0-  Feb. 8, 2015, 2:11 a.m. — [ratchode](https://www.bashoneliners.com/oneliners/users/212/) | |

|  |  |
| --- | --- |
| 1 score | [**Download a file from a webserver with telnet**](https://www.bashoneliners.com/oneliners/211/)   $ (echo 'GET /'; echo; sleep 1; ) | telnet www.google.com 80  Dec. 22, 2014, 11:31 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |
| 0 score | [**Print the window title of current mpv session to display what is playing**](https://www.bashoneliners.com/oneliners/210/)   $ wmctrl -pl | grep $(pidof mpv) | cut -d- -f2-  Dec. 15, 2014, 3:37 a.m. — [openiduser171](https://www.bashoneliners.com/oneliners/users/208/) | |

|  |  |  |
| --- | --- | --- |
| 4 score | [**Change the encoding of all files in a directory and subdirectories**](https://www.bashoneliners.com/oneliners/209/)   $ find . -type f -name '\*.java' -exec sh -c 'iconv -f cp1252 -t utf-8 "$1" > converted && mv converted "$1"' -- {} \;  Nov. 20, 2014, 12:15 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) | |
| 3 score | [**Generate a sequence of numbers**](https://www.bashoneliners.com/oneliners/208/)   $ for ((i=1; i<=10; ++i)); do echo $i; done  Nov. 4, 2014, 12:29 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |

|  |  |
| --- | --- |
| 0 score | [**Shuffle lines**](https://www.bashoneliners.com/oneliners/207/)   $ ... | perl -MList::Util -e 'print List::Util::shuffle <>'  Oct. 25, 2014, 10:40 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |
| 1 score | [**Print a flat list of dependencies of a Maven project**](https://www.bashoneliners.com/oneliners/206/)   $ mvn dependency:list | sed -ne s/..........// -e /patterntoexclude/d -e s/:compile//p -e s/:runtime//p | sort | uniq  Sept. 22, 2014, 9:02 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) | |

|  |  |  |
| --- | --- | --- |
| 1 score | [**Open Windows internet shortcut (\*.url) files in firefox**](https://www.bashoneliners.com/oneliners/205/)   $ grep -i url='\*' file.url | cut -b 5- | xargs firefox  Sept. 12, 2014, 12:06 a.m. — [tsjswimmer](https://www.bashoneliners.com/oneliners/users/201/) | |
| 0 score | [**Open Windows internet shortcut (\*.url) files in firefox**](https://www.bashoneliners.com/oneliners/204/)   $ firefox $(grep -i ^url='\*' file.url | cut -b 5-)  Sept. 11, 2014, 10:03 a.m. — [tsjswimmer](https://www.bashoneliners.com/oneliners/users/201/) |

|  |  |
| --- | --- |
| 1 score | [**Remove all at jobs**](https://www.bashoneliners.com/oneliners/203/)   $ atq | sed 's\_\([0-9]\{1,8\}\).\*\_\1\_g' | xargs atrm |

* [Tags](https://www.bashoneliners.com/oneliners/tags/)
* [Popular](https://www.bashoneliners.com/oneliners/popular/)
* [Active](https://www.bashoneliners.com/oneliners/active/)
* [Newest](https://www.bashoneliners.com/oneliners/newest/)

|  |  |
| --- | --- |
| 5 score | [**Corporate random bullshit generator (cbsg)**](https://www.bashoneliners.com/oneliners/201/)   $ curl -s http://cbsg.sourceforge.net/cgi-bin/live | grep -Eo '^<li>.\*</li>' | sed s,\</\\?li\>,,g | shuf -n 1  Sept. 4, 2014, 3:44 p.m. — [Genunix](https://www.bashoneliners.com/oneliners/users/197/) |
| 1 score | [**Deletes orphan vim undo files**](https://www.bashoneliners.com/oneliners/200/)   $ find . -type f -iname '\*.un~' | while read UNDOFILE ; do FILE=$( echo "$UNDOFILE" | sed -r -e 's/.un~$//' -e 's&/\.([^/]\*)&/\1&' ) ; [[ -e "$FILE" ]] || rm "$UNDOFILE" ; done  Sept. 2, 2014, 6:51 p.m. — [rafaeln](https://www.bashoneliners.com/oneliners/users/196/) |

|  |  |
| --- | --- |
| 2 score | [**Generates random texts**](https://www.bashoneliners.com/oneliners/199/)   $ tr -dc a-z1-4 </dev/urandom | tr 1-2 ' \n' | awk 'length==0 || length>50' | tr 3-4 ' ' | sed 's/^ \*//' | cat -s | fmt  July 31, 2014, 10:45 p.m. — [bkmeneguello](https://www.bashoneliners.com/oneliners/users/189/) |
| 2 score | [**Find recent logs that contain the string "Exception"**](https://www.bashoneliners.com/oneliners/198/)   $ find . -name '\*.log' -mtime -2 -exec grep -Hc Exception {} \; | grep -v :0$  July 19, 2014, 7:53 a.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |

|  |  |
| --- | --- |
| 2 score | [**Parse nginx statistics output**](https://www.bashoneliners.com/oneliners/197/)   $ i=$(curl -s server/nginx\_stats); IFS=$'\n'; i=($i); a=${i[0]/Active connections: } && a=${a/ }; r=${i[2]# [0-9]\* [0-9]\* }; echo "Active: $a, requests: $r"  June 20, 2014, 3:19 p.m. — [azat](https://www.bashoneliners.com/oneliners/users/185/) |
| 2 score | [**Install profiling versions of all libghc dpkg packages**](https://www.bashoneliners.com/oneliners/196/)   $ sudo dpkg -l | grep libghc | grep "\-dev" | cut -d " " -f 3 | tr '\n' ' ' | sed -e 's/\-dev/\-prof/g' | xargs sudo apt-get install --yes  May 26, 2014, 1:14 a.m. — [openiduser146](https://www.bashoneliners.com/oneliners/users/181/) |

|  |  |
| --- | --- |
| 3 score | [**Compute factorial of positive integer**](https://www.bashoneliners.com/oneliners/195/)   $ fac() { (echo 1; seq $1) | paste -s -d\\* | bc; }  May 21, 2014, 10:55 p.m. — [jeroenjanssens](https://www.bashoneliners.com/oneliners/users/179/) |
| 1 score | [**Extensive "cleanup" operations following "sudo yum upgrade"**](https://www.bashoneliners.com/oneliners/194/)   $ sudo yum upgrade && for pkg in $(package-cleanup --orphans -q); do repoquery $(rpm -q $pkg --queryformat="%{NAME}") | grep -q ".\*" && echo $pkg; done | xargs sudo yum -y remove && for pkg in $(package-cleanup --leaves --all -q); do repoquery --groupmember $pkg | grep -q "@" || echo $pkg; done  April 16, 2014, 9:58 p.m. — [openiduser143](https://www.bashoneliners.com/oneliners/users/177/) | |

|  |  |
| --- | --- |
| 3 score | [**Find all files recursively with specified string in the filename and output any lines found containing a different string.**](https://www.bashoneliners.com/oneliners/193/)   $ find . -name \*conf\* -exec grep -Hni 'matching\_text' {} \; > matching\_text.conf.list  April 14, 2014, 8:23 p.m. — [n00tz](https://www.bashoneliners.com/oneliners/users/148/) |
| 7 score | [**Displays the quantity of connections to port 80 on a per IP basis**](https://www.bashoneliners.com/oneliners/192/)   $ clear;while x=0; do clear;date;echo "";echo " [Count] | [IP ADDR]";echo "-------------------";netstat -np|grep :80|grep -v LISTEN|awk '{print $5}'|cut -d: -f1|uniq -c; sleep 5;done  April 9, 2014, 5:49 a.m. — [cesp](https://www.bashoneliners.com/oneliners/users/176/) |

|  |  |
| --- | --- |
| 1 score | [**Get average CPU temperature from all cores.**](https://www.bashoneliners.com/oneliners/191/)   $ \_\_=`sensors | grep Core` && echo \(`echo $\_\_ | sed 's/.\*+\(.\*\).C\(\s\)\+(.\*/\1/g' | tr "\n" "+" | head -c-1`\)\/`echo $\_\_ | wc -l` | bc && unset \_\_  April 2, 2014, 10:04 p.m. — [openiduser139](https://www.bashoneliners.com/oneliners/users/173/) |
| 1 score | [**Concatenate multiple SSL certificate files to make one PEM file**](https://www.bashoneliners.com/oneliners/190/)   $ files=("yourcert.crt" "provider.ca.pem") && for i in ${files[@]} ; do $(cat $i >> yourcert.pem && echo "" >> yourcert.pem) ; done  April 2, 2014, 5:41 p.m. — [renoirb](https://www.bashoneliners.com/oneliners/users/172/) |

|  |  |  |
| --- | --- | --- |
| 1 score | [**List all non Git comited files and make a gzip archive with them**](https://www.bashoneliners.com/oneliners/189/)   $ GITFOLDER="/srv/some/folder" ls-files --others --exclude-standard | tar czf ${GITFOLDER}-archives/uploads-$(date '+%Y%m%d%H%M').tar.gz -T -  April 2, 2014, 5:18 p.m. — [renoirb](https://www.bashoneliners.com/oneliners/users/172/) | |
| 2 score | [**Have script run itself in a virtual terminal**](https://www.bashoneliners.com/oneliners/188/)   $ tty >/dev/null || { urxvt -hold -e "$0" "$@" & exit; }  March 6, 2014, 3:18 a.m. — [openiduser111](https://www.bashoneliners.com/oneliners/users/143/) |

|  |  |  |
| --- | --- | --- |
| 11 score | [**Show 10 Largest Open Files**](https://www.bashoneliners.com/oneliners/187/)   $ lsof / | awk '{ if($7 > 1048576) print $7/1048576 "MB" " " $9 " " $1 }' | sort -n -u | tail  Feb. 28, 2014, 3:34 a.m. — [cellojoe](https://www.bashoneliners.com/oneliners/users/164/) | |
| 3 score | [**Extract your external IP address using dig**](https://www.bashoneliners.com/oneliners/186/)   $ dig +short myip.opendns.com @resolver1.opendns.com  Feb. 25, 2014, 7:50 a.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |

|  |  |  |
| --- | --- | --- |
| 2 score | [**Remove .DS\_Store from the repository you happen to staging by mistake**](https://www.bashoneliners.com/oneliners/185/)   $ find . -name .DS\_Store -exec git rm --ignore-unmatch --cached {} +  Feb. 22, 2014, 9:45 a.m. — [Kuwana](https://www.bashoneliners.com/oneliners/users/163/) | |
| 3 score | [**Remove offending key from known\_hosts file with one swift move**](https://www.bashoneliners.com/oneliners/184/)   $ ssh-keygen -R <hostname>  Jan. 25, 2014, 1:35 p.m. — [openiduser126](https://www.bashoneliners.com/oneliners/users/159/) |

|  |  |  |
| --- | --- | --- |
| 0 score | [**Check if a file exists and has a size greater than X**](https://www.bashoneliners.com/oneliners/183/)   $ [[ $(find /path/to/file -type f -size +51200c 2>/dev/null) ]] && echo true || echo false  Jan. 9, 2014, 12:34 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) | |
| 1 score | [**Converts DD/MM/YYYY date format to ISO-8601 (YYYY-MM-DD)**](https://www.bashoneliners.com/oneliners/182/)   $ sed 's\_\([0-9]\{1,2\}\)/\([0-9]\{1,2\}\)/\([0-9]\{4\}\)\_\3-\2-\1\_g'  Dec. 30, 2013, 10:30 a.m. — [laurip](https://www.bashoneliners.com/oneliners/users/158/) |

|  |  |  |
| --- | --- | --- |
| 0 score | [**Replace sequences of the same characters with a single character**](https://www.bashoneliners.com/oneliners/181/)   $ echo heeeeeeelllo | sed 's/\(.\)\1\+/\1/g'  Dec. 11, 2013, 7:58 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) | |
| 1 score | [**Counting the number of commas in CSV format**](https://www.bashoneliners.com/oneliners/180/)   $ perl -ne 'print tr/,//, "\n"' < file.csv | sort -u  Dec. 1, 2013, 1:03 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |

|  |  |  |
| --- | --- | --- |
| 0 score | [**Count the lines of each file extension in a list of files**](https://www.bashoneliners.com/oneliners/179/)   $ git ls-files | xargs wc -l | awk -F ' +|\\.|/' '{ sumlines[$NF] += $2 } END { for (ext in sumlines) print ext, sumlines[ext] }'  Nov. 9, 2013, 11:49 a.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) | |
| 0 score | [**Add all unknown files in a Subversion checkout**](https://www.bashoneliners.com/oneliners/178/)   $ svn add . --force  Sept. 24, 2013, 7:59 a.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |

|  |  |
| --- | --- |
| 0 score | [**Find files that are not executable**](https://www.bashoneliners.com/oneliners/177/)   $ find /some/path -type f ! -perm -111 -ls  Sept. 18, 2013, 9:14 p.m. — [Janos](https://www.bashoneliners.com/oneliners/users/8/) |