**How To Find Files by Content Under UNIX / Linux**

Author: Vivek Gite Last updated: July 12, 2022 [26 comments](https://www.cyberciti.biz/faq/unix-linux-finding-files-by-content/#comments)



I had written lots of code in C for my school work and saved it as source code under /home/user/c/\*.c and \*.h. How do I find files by content such as string or words (function name such as main() under UNIX shell prompt?  
  
You need to use the following tools to find files by content under Unix or Linux operating systems:

Advertisement

1. [**grep command**](https://www.cyberciti.biz/faq/howto-use-grep-command-in-linux-unix/?utm_source=Linux_Unix_Command&utm_medium=faq&utm_campaign=nixcmd) : print lines matching a pattern.
2. **find command** : search for files in a directory hierarchy.

**Using**[**grep Command To Find Files By**](https://www.cyberciti.biz/faq/howto-search-find-file-for-text-string/)**Content on Unix or Linux**

Type the command as follows:

grep 'string' \*.txt

grep 'main(' \*.c

grep '#include<example.h>' \*.c

grep 'getChar\*' \*.c

grep -i 'ultra' \*.conf

grep -iR 'ultra' \*.conf

Where

* **-i**: Ignore case distinctions in both the PATTERN (match valid, VALID, ValID string) and the input files (math file.c FILE.c FILE.C filename).
* **-R** (or **-r**): Read all files under each directory, recursively.

You can also use the [egrep command](https://www.cyberciti.biz/faq/grep-regular-expressions/?utm_source=Linux_Unix_Command&utm_medium=faq&utm_campaign=nixcmd" \o "Regular expressions in grep ( regex ) with examples). For example, find two words named foo and bar in all conf files:

$ egrep 'foo|bar' \*.conf

**Finding files by content using**[**shell pipes**](https://bash.cyberciti.biz/guide/Pipes)

Both [grep command](https://www.cyberciti.biz/faq/howto-use-grep-command-in-linux-unix/?utm_source=Linux_Unix_Command&utm_medium=faq&utm_campaign=nixcmd) or [egrep command](https://www.cyberciti.biz/faq/grep-regular-expressions/?utm_source=Linux_Unix_Command&utm_medium=faq&utm_campaign=nixcmd" \o "Regular expressions in grep ( regex ) with examples) works on input provided from files or stdin or shell pipes. For instance, run the ls command and search for resume.doc and wife-birthday-party.png using the following syntax:

$ ls -l | egrep 'resume.doc|wife-birthday-party.png'  
$ ls -l | grep -i -E 'resume.doc|wife-birthday-party.png'

**Highlighting searched patterns**

You can highlight patterns easily while searching large number of files by passing the --color option as follows:

$ grep --color=auto -iR 'getChar();' \*.c

**Displaying file names and line number for searched patterns**

You may also need to display filenames and numbers, hence pass the -H option:

$ grep --color=auto -iRnH 'getChar();' \*.c

Where,

* **-n** : Prefix each line of output with the 1-based line number within its input file.
* **-H** Print the file name for each match. This is the default when there is more than one file to search.

Next, type the following command:

$ grep --color=auto -nH 'WORD\_HERE' \*  
$ grep --color=auto -nH 'DIR' \*

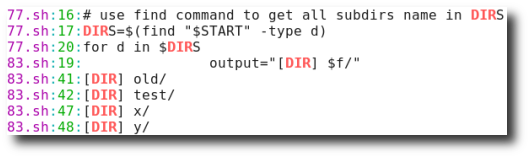
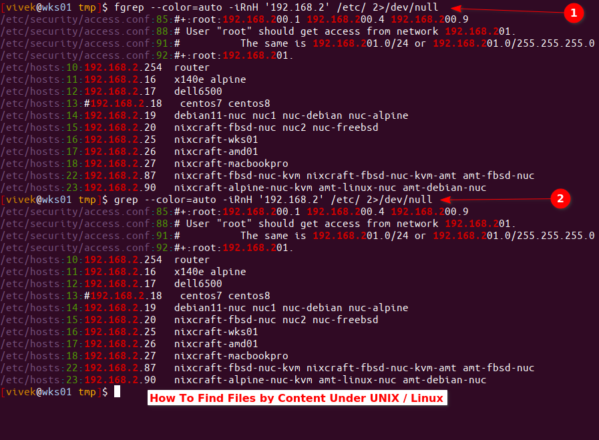


Fig.01: grep command displaying searched pattern

In this example, I want to find a string named ‘192.168.2’ for /etc/ directory:

$ fgrep --color=auto -iRnH '192.168.2' /etc/ 2>/dev/null  
$ grep --color=auto -iRnHf '192.168.2' /etc/ 2>/dev/null



Click to enlarge

The fgrep or -f option passed to the grep interprets PATTERNS as fixed strings, not regular expressions (regex). The 2>/dev/null used to hide permission error message by [sending stderr to /dev/null](https://www.cyberciti.biz/faq/how-to-redirect-output-and-errors-to-devnull/).

**Using find command to search files by words or string**

We can also use the find command. The syntax is as follows:

$ find /dir/to/search -name "file-pattern" -print | xargs grep "word-to-search"  
## OR ##  
$ find /dir/to/search -iname "file-pattern" -print0 | xargs -I {} -0 grep "string-to-search" "{}"

For example, search all c program files (\*.c) and look for “main(” and print it on the screen when matched in the current directory:

$ find . -name "\*.c" -print | xargs grep "main("

OR

$ find /projects/ -iname "\*.c" -print0 | xargs -I {} -0 grep "main(" "{}"

Where find command options are:

* **-name** : Base of file name. For instance, look for all Perl files (\*.pl)
* **-iname** : Same as above (-name
* **-print** : Print the full file name on the standard output.
* **-print0** : Display the full file name on the standard output, followed by a null character (instead of the newline character that -print uses). This allows file names that contain newlines or other types of white space to be correctly interpreted by programs that process the find output. This option corresponds to the -0 option of xargs.

And, the xargs command are:

* **-I {}** : Replace occurrences of {} in the initial-arguments with names read from standard input. In other words, pass {} as input to the grep command.
* **-0** : Input items are terminated by a null character instead of by whitespace, and the quotes and backslash are not special (every character is taken literally). Disables the end of file string, which is treated like any other argument. Useful when input items might contain white space, quote marks, or backslashes. The GNU find -print0 option produces input suitable for this mode.
* **grep "main(" "{}"** : Search for "main(" string using the grep in a file found by find command.

**Conclusion**

You learned how to find files by content under UNIX and Linux using various commands. See the following resources:

* $ man grep  
  $ man find  
  $ grep --help

This entry is **6** of **7** in the **Linux / UNIX grep Command Tutorial** series. Keep reading the rest of the series: