English | Deutsch Log in or Sign up

Tutorials Tags Forums Linux Commands Subscribe ISPConfig News

Tutorial search

Tutorials How to use grep to search for strings in files on the shell

1 The GREP command- an overview

The grep command, which means **global regular expression print**, remains amongst the most versatile commands in a Linux terminal environment. It happens to be an immensely powerful program that lends users the ability to sort input based on complex rules, thus rendering it a fairly popular link across numerous command chains. The grep command is primarily used to search text or search any given file for lines containing a match to the supplied words/strings. By default, grep displays the matching lines, and it may be used to search for lines of text matching one/many regular expressions in a fuss-free, and it outputs only the matching lines.

2 The basic grep command syntax

The basic grep command syntax is as follows:

```
grep 'word' filename
grep 'word' file1 file2 file3
grep 'string1 string2'  filename
cat otherfile | grep 'something'
command | grep 'something'
command option1 | grep 'data'
grep --color 'data' fileName
```

On this page

3 How to use the grep command for searching in a file

In the first example, I will search for the user "tom" in the Linux passwd file. To search the /etc/passwd file for the user "tom", you need to enter the following command:

```
grep tom /etc/passwd
```

Given below are the sample Outputs:

```
tom:x:1000:1000:tom,,,:/home/tom:/bin/bash
```

You have the option to instruct grep to ignore word case, i.e., match abc, Abc, ABC and all possible combinations with the -i option) as shown below:

```
grep -i "tom" /etc/passwd
```

4 Recursive use of grep

If you have a bunch of text files in a directory heirarchy, e.g, the apache configuration files in /etc/apache2/ and you want to find the file where a specific text is defined, then use the -r option of the grep command to do a recursive search meaning they can read all files under each directory for a string "197.167.2.9" (as shown below):

```
grep -r "mydomain.com" /etc/apache2/
```

Alternatively, the following command may be used:

```
grep -R "mydomain.com" /etc/apache2/
```

Given below are the Sample outputs for a similar search on a nginx server:

```
grep -r "mydomain.com" /etc/nginx/
/etc/nginx/sites-available/mydomain.com.vhost:        if ($http_host != "www.mydomain.com") {
```

Here, you would see the result for mydomain.com on a distinct line preceded by the name of the file (for instance /etc/nginx/sites-available/mydomain.com.vhost) in which it was found. The inclusion of the file names in the output data may be easily suppressed by using the -h option (as explained below): grep -h -R "mydomain.com" /etc/nginx/ Given below are the Sample Outputs:

```
grep -r "mydomain.com" /etc/nginx/
if ($http_host != "www.mydomain.com") {
```

5 Using grep to search only for words

When you are searching for abc, grep will match all sorts of things, viz., kbcabc, abc123, aarfbc35 and lots more combinations. You can compel the grep command to select only those lines that contain matches to form whole words (those that match only abc word), as shown below:

```
grep -w "abc" file
```

6 Using grep to search two different words

To search for two different words, you must use the egrep command as shown below:

```
egrep -w 'word1|word2' /path/to/file
```

7 Count line for matched words

The grep command has the ability to report the number of times a particular pattern has been matched for each file using the -c (count) option (as shown below):

```
grep -c 'word' /path/to/file
```

In addition, users may use the -n option preceding each output line with the number of the line in the text file from which it was obtained (as shown below):

```
grep -n 'root' /etc/passwd
```

Given below are the Sample outputs:

```
1:root:x:0:0:root:/root:/bin/bash
```

8 Grep invert match

Users may make use of the -v option to print inverts the match, which means it would match only those lines that do not contain the given word. For instance, print all lines that do not contain the word par by using the following command:

```
grep -v par /path/to/file
```

9 How to list only the names of matching files

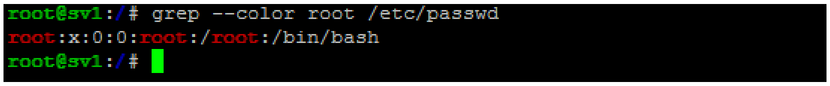
You must use the -l option to list file name whose contents mention a particular word, for instance primary, using the following command:

```
grep -l 'primary' *.c
```

Lastly, you have the option to compel grep to display output in specific colors by using the following command:

```
grep --color root /etc/passwd
```

Given below are Sample Outputs:



```
root@sv1:/# grep --color root /etc/passwd
root:x:0:0:root:/root:/bin/bash
root@sv1:/#
```

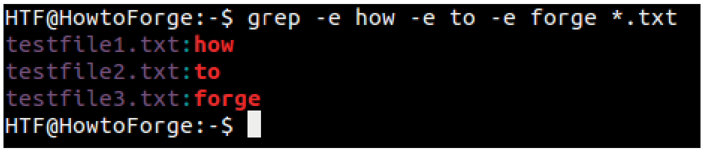
10 How to make grep command handle multiple search patterns?

There could be situations where-in you might want to search multiple patterns in a given file (or set of files). In such scenarios, you should use the -e command line option that grep provides.

For example, suppose you want to search for words "how", "to", and "forge" in all the text files present in your current working directory, then here's how you can do this:

```
grep -e how -e to -e forge *.txt
```

Here's the command in action:



```
HTF@HowtoForge:-$ grep -e how -e to -e forge *.txt
testfile1.txt:how
testfile2.txt:to
testfile3.txt:forge
HTF@HowtoForge:-$
```

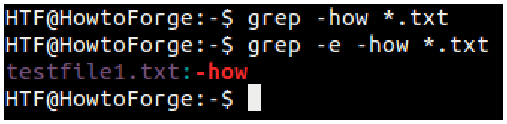
The -e command line option also helps in scenarios where-in the pattern begins with a hyphen (-). For example, if you want to search for, say, "-how", then the following command won't be helpful:

```
grep -how *.txt
```

It's when you use the -e command line option, the command understands what exactly you are trying to search in this case:

```
grep -e -how *.txt
```

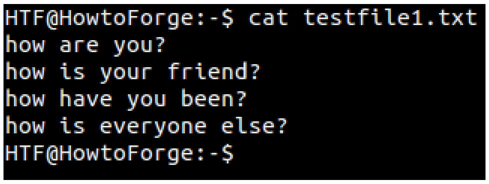
Here are both commands in action:



```
HTF@HowtoForge:~$ grep -how *.txt
HTF@HowtoForge:~$ grep -e -how *.txt
testfile1.txt:-how
HTF@HowtoForge:~$
```

11 How to limit grep output to a particular number of lines?

In case you want to limit the grep output to a particular number of lines, you can do that using the -m command line option. For example, suppose you want to search for the word "how" in testfile1.txt which contains the following lines:

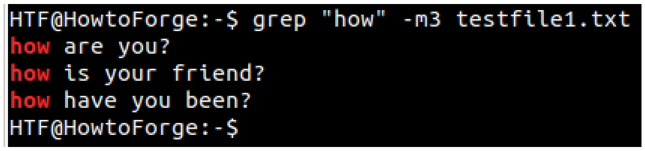


```
HTF@HowtoForge:~$ cat testfile1.txt
how are you?
how is your friend?
how have you been?
how is everyone else?
HTF@HowtoForge:~$
```

But the requirement is for grep to stop searching after 3 lines containing the searched pattern have been found. So, to do this, you can run the following command:

```
grep "how" -m3 testfile1.txt
```

Here's the command in action:



```
HTF@HowtoForge:~$ grep "how" -m3 testfile1.txt
how are you?
how is your friend?
how have you been?
HTF@HowtoForge:~$
```

Moving on, following is what the command's man page says:

```
If the input is standard input from a regular file, and NUM matching lines are output, grep ensuresthat the standard input is positioned to just after the last matching line before exiting, regardless of the presence of trailing context lines. This enables a calling process to resume a search.
```

So for example, if you have a bash script that has a loop, and you want to fetch one match per loop iteration, then using 'grep -m1' will do the needful.

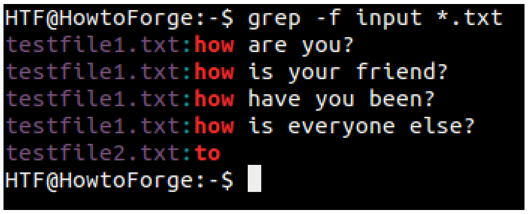
12 How to make grep obtain patterns from file?

If you want, you can also make the grep command obtain patterns from a file. The tool's -f command line option lets you do this.

For example, suppose you want to search all the .txt files in the current directory for words "how" and "to", but want to supply these input strings through a file named, say, "input," then here's how you can do this:

```
grep -f input *.txt
```

Here's the command in action:

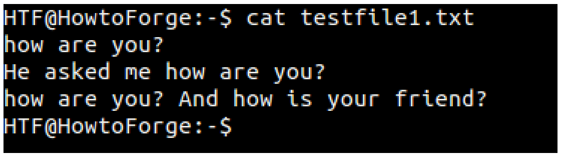


```
HTF@HowtoForge:~$ grep -f input *.txt
testfile1.txt:how are you?
testfile1.txt:how is your friend?
testfile1.txt:how have you been?
testfile1.txt:how is everyone else?
testfile2.txt:to
HTF@HowtoForge:~$ 
```

13 How to make grep display only those lines that completely match search pattern

Up until now, we have seen that by default grep matches and displays complete lines that contain search pattern. But if the requirement is to make grep only display those lines that completelty match the searched pattern, then this can be done using the -x command line option.

For example, suppose testfile1.txt file contains the following lines:

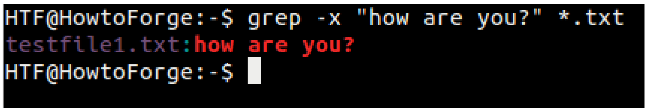


```
HTF@HowtoForge:~$ cat testfile1.txt
how are you?
He asked me how are you?
how are you? And how is your friend?
HTF@HowtoForge:~$
```

And the pattern you want to search is "how are you?". So to make sure that grep only displays lines that completely match this pattern, use it in the following way:

```
grep -x "how are you?" *.txt
```

Here's the command in action:



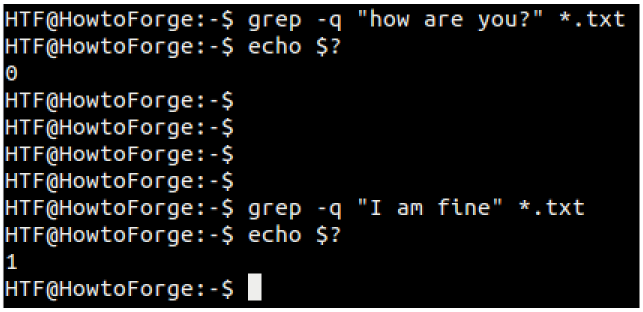
```
HTF@HowtoForge:~$ grep -x "how are you?" *.txt
testfile1.txt:how are you?
HTF@HowtoForge:~$ 
```

14 How to force grep to not display anything in output

There might be situations where-in you don't need the grep command to produce anything in the output. Instead, you just want to know whether or not a match was found based on the command's exit status. This can be achieved using the -q command line option.

While the -q option mutes the output, the tool's exit status can be confirmed by the 'echo $?' command. In case of grep, the command exits with '0' status when it's successful (meaning, a match was found), while it exits with status '1' when no match was found.

The following screenshots shows both the successful and unsuccessful scenarios:



```
HTF@HowtoForge:~$ grep -q "how are you?" *.txt
HTF@HowtoForge:~$ echo $?
0
HTF@HowtoForge:~$
HTF@HowtoForge:~$
HTF@HowtoForge:~$
HTF@HowtoForge:~$
HTF@HowtoForge:~$ grep -q "I am fine" *.txt
HTF@HowtoForge:~$ echo $?
1
HTF@HowtoForge:~$ 
```

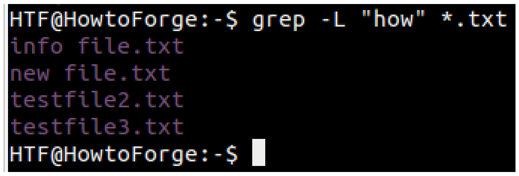
15 How to make grep display name of files that do not contain search pattern?

By default, the grep command displays name of files containing the search pattern (as well as matched lines). This is quite logical, as that's what expected of this tool. However, there might be cases where-in the requirement could be to get names of those files that do not contain the searched pattern.

This is also possible with grep - the -L options lets you do this. So, for example, to find all those text files in the current directory that do not contain the word "how", you can run the following command:

```
grep -L "how" *.txt
```

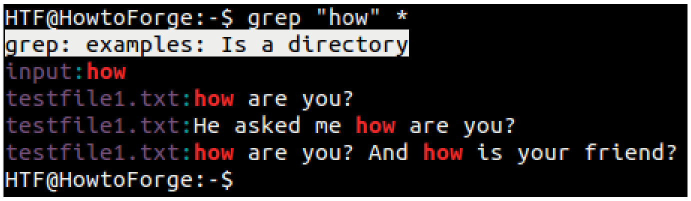
Here's the command in action:



```
HTF@HowtoForge:~$ grep -L "how" *.txt
info file.txt
new file.txt
testfile2.txt
testfile3.txt
HTF@HowtoForge:~$
```

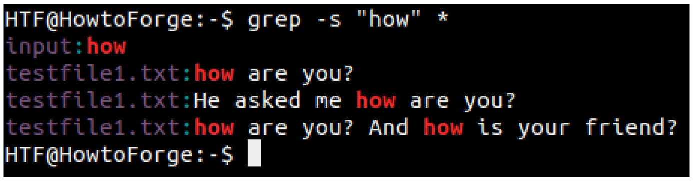
16 How to suppress error messages produced by grep?

If you want, you can also force grep to mute any error messages it displays in the output. This can be done using the -s command line option. For example, consider the following scenario in which grep produces error/warning related to the directory it encounters:



```
HTF@HowtoForge:~$ grep "how" *
grep: examples: Is a directory
input:how
testfile1.txt:how are you?
testfile1.txt:He asked me how are you?
testfile1.txt:how are you? And how is your friend?
HTF@HowtoForge:~$
```

So in these kind of scenarios, the -s command line option helps. See below.



```
HTF@HowtoForge:~$ grep -s "how" *
input:how
testfile1.txt:how are you?
testfile1.txt:He asked me how are you?
testfile1.txt:how are you? And how is your friend?
HTF@HowtoForge:~$
```

So you can see that the error/warning got muted.

17 How to make grep recursively search directories?

As clear from the example used in the previous point, the grep command doesn't do a recursive search by default. To make sure your grep search is recursive, use the -d command line option and pass the value 'recurse' to it.

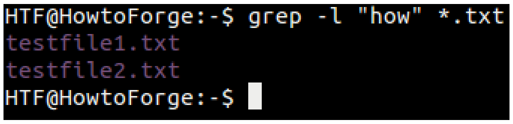
```
grep -d recurse "how" *
```

Note1: The directory related error/warning message we discussed in the previous point can also be muted using the -d option - all you have to do is to pass the value 'skip' to it.

Note2: Use '--exclude-dir=[DIR]' option to exclude directories matching the pattern DIR from recursive searches.

18 How to make grep terminate file names with NULL character?

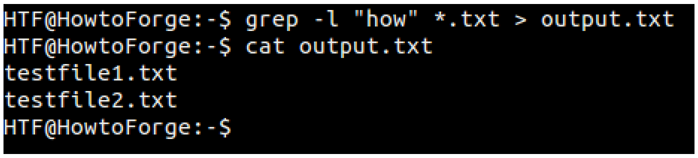
As we have already discussed, the -l command line option of grep is used when you only want the tool to display filenames in the output. For example:



```
HTF@HowtoForge:-$ grep -l "how" *.txt
testfile1.txt
testfile2.txt
HTF@HowtoForge:-$
```

Now, what you should know here is that each name in the above output is separated/terminated by a newline character. Here's how you can verify that:

Redirect the output to a file, and the print the file contents:



```
HTF@HowtoForge:-$ grep -l "how" *.txt > output.txt
HTF@HowtoForge:-$ cat output.txt
testfile1.txt
testfile2.txt
HTF@HowtoForge:-$
```

So the output of the cat command confirms the presence of a newline character between the file names.

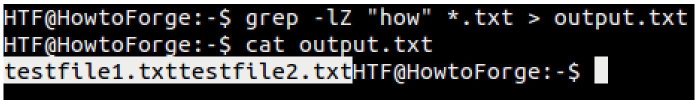
But as you might already know, the newline character can be part of a file name as well. So when dealing with cases where-in filenames contain newline and they are separated/terminated by newline as well, it becomes difficult to work on the grep output (especially when accessing the output through a script).

It would be good if the separating/terminating character is not newline. Well, you'll be glad to know that grep provides a command line option -Z that makes sure filenames are followed by a NULL character and not a newline.

So, in our case, the command becomes:

```
grep -lZ "how" *.txt
```

Here's how we confirmed the presence of NULL character:



```
HTF@HowtoForge:-$ grep -lZ "how" *.txt > output.txt
HTF@HowtoForge:-$ cat output.txt
testfile1.txttestfile2.txtHTF@HowtoForge:-$
```

Following is a related command line option that you should know:

```
 -z, --null-data
Treat the input as a set of lines, each terminated by a zero byte (the ASCII NUL character) insteadof a newline.
 Like the -Z or --null option, this option can be used with commands like sort -z to process arbitrary file name
s.
```

19 More GREP command examples

In our second GREP command tutorial, you can find even more examples on how to use this Linux command.

- How to perform pattern search in files using Grep

Suggested articles

5 Comment(s)

Add comment

Name *

Email *

B *I*

p

□ I'm not a robot

reCAPTCHA
Privacy - Terms

Submit comment

Comments

From: harshal sarode **at:** *2015-06-22 10:55:54* Reply

hi sir,
i am facing a problm that how to fetch the common words from a file by grep.i tried thro the hardcoded string like
grep "abc" MyFile.txt.if i dont know if there are any common words in the file or not.then how i do it?plz help me so.

From: ramakrishna **at:** *2015-07-17 12:10:44* Reply

Hi Sir,
I wants to get the content in beetween the particulers word like
It starts with subject and ends with subject and i wants the content in between that

```
find . -exec grep -H "" {} \;
```

From: dkolb001 **at:** *2016-06-23 15:46:33* Reply

I have two files with content. file1 has two columns of content, and file2 has a single column of content.
I would like to use grep to find all matching rows of file1 content, and that matches file2 row of content and display.
Not all content has a match for file1 and file2, but I would like the match to be correct.
Any advise for this?

From: Daniel **at:** *2016-07-08 07:33:08* Reply

I'm a linux novice and im using the grep function to search for the number 1. "grep 1 tdocs" now its working it's giving me car1 wheel1 but then also car11 and 12. How do i go about specifing the search so it's just car1 and wheel1? Many Thanks.

From: Brandon **at:** *2016-12-04 16:53:46* Reply

I need some help with grep...
I have two .txt files that contain lists, the first, a.txt, contains:
abcd
and the other, b.txt, contains
abgcd
I want to use grep to output items from b.txt that arent in a.txt, in this case, "g"
I think it goes something like

```
grep -v --file=a.txt b.txt
```

Tutorials › **How to use grep to search for strings in files on the shell**

Sign up now!

Tutorial Info

Author:	srijan
Last updated:	Mar 30, 2017
Tags:	
	centos, debian, linux, opensuse, shell, ubuntu

Share This Page

Tweet Follow 27.8K followers

Recommend 24

G+1 13

Contribute Contact Help Imprint

Terms