Find Command

find command in Linux with examples

The **find** command in UNIX is a command line utility for walking a file hierarchy. It can be used to find files and directories and perform subsequent operations on them. It supports searching by file, folder, name, creation date, modification date, owner and permissions. By using the ‘-exec’ other UNIX commands can be executed on files or folders found.

**Syntax :**

$ find [where to start searching from]

[expression determines what to find] [-options] [what to find]

**Options :**

* **-exec CMD:** The file being searched which meets the above criteria and returns 0 for as its exit status for sucessful command execution.
* **-ok CMD :** It works same as -exec except the user is prompted first.
* **-inum N :** Search for files with inode number ‘N’.
* **-links N :** Search for files with ‘N’ links.
* **-name demo :** Search for files that are specified by ‘demo’.
* **-newer file :** Search for files that were modified/created after ‘file’.
* **-perm octal :** Search for the file if permission is ‘octal’.
* **-print :** Display the path name of the files found by using the rest of the criteria.
* **-empty :** Search for empty files and directories.
* **-size +N/-N :** Search for files of ‘N’ blocks; ‘N’ followed by ‘c’can be used to measure size in characters; ‘+N’ means size > ‘N’ blocks and ‘-N’ means size < 'N' blocks.
* **-user name :** Search for files owned by user name or ID ‘name’.
* **\(expr \) :** True if ‘expr’ is true; used for grouping criteria combined with OR or AND.
* **! expr :** True if ‘expr’ is false.

**1. Search a file with specific name.**

$ find ./GFG -name sample.txt

# find . -name tecmint.txt

./tecmint.txt

# find /home -name tecmint.txt

/home/tecmint.txt

--find file ignoring case

# find /home -iname tecmint.txt

./tecmint.txt

./Tecmint.txt

-- find directories

# find / -type d -name Tecmint

/Tecmint

--find files

# find . -type f -name tecmint.php

./tecmint.php

**2. Search a file with pattern.**

$ find ./GFG -name \*.txt

# find . -type f -name "\*.php"

./tecmint.php

./login.php

./index.php

**3. How to find and delete a file with confirmation.**

$ find ./GFG -name sample.txt -exec rm -i {} \;

find . -name "\*.bak" -delete

**4. Search for empty files and directories.**

$ find ./GFG -empty

**5. Search for file with entered permissions.**

$ find ./GFG -perm 664

# find . -type f -perm 0777 –print

# find / -type f ! -perm 777

--find read only files

# find / -perm /u=r

--- find executable files

# find / -perm /a=x

---Find Files with 777 Permissions and Chmod to 644

# find / -type d -perm 777 -print -exec chmod 755 {} \;

**6. Search text within multiple files.**

$ find ./ -type f -name "\*.txt" -exec grep 'Geek' {} \;

Before the implementation of the -exec option, this kind of command might have used the xargscommand to generate a similar output:

$ find . -type f -print | xargs grep "example"

**7.  Find Particular Files of User**

# find /home -user tecmint -iname "\*.txt"

**8. Find Last 50 Days Modified Files**

# find / -mtime 50 : modified

# find / -atime 50 : access

**9. Find Last 50-100 Days Modified Files**

# find / -mtime +50 –mtime -100

**10. Find Changed Files in Last 1 Hour**

# find / -cmin -60

**11. Find Modified Files in Last 1 Hour**

# find / -mmin -60

**12. Find Accessed Files in Last 1 Hour**

# find / -amin -60

**13. Find 50MB Files**

# find / -size 50M

**14. Find Size between 50MB – 100MB**

# find / -size +50M -size -100M

**15. Find and Delete 100MB Files**

# find / -size +100M -exec rm -rf {} \;

--- Xargs

## How to use xargs

By default xargs reads items from standard input as separated by blanks and executes a command once for each argument. In the following example standard input is piped to xargs and the mkdir command is run for each argument, creating three folders.

echo 'one two three' | xargs mkdir

ls

one two three

## How to use xargs with find

The most common usage of xargs is to use it with the [find](https://shapeshed.com/unix-find/) command. This uses find to search for files or directories and then uses xargs to operate on the results. Typical examples of this are removing files, changing the ownership of files or moving files.

find and xargs can be used together to operate on files that match certain attributes. In the following example files older than two weeks in the temp folder are found and then piped to the xargs command which runs the rmcommand on each file and removes them.

find /tmp -mtime +14 | xargs rm

## xargs v exec {}

The find command supports the -exec option that allows arbitrary commands to be found on files that are found. The following are equivalent.

find ./foo -type f -name "\*.txt" -exec rm {} \;

find ./foo -type f -name "\*.txt" | xargs rm

So which one is faster? Let’s compare a folder with 1000 files in it.

time find . -type f -name "\*.txt" -exec rm {} \;

0.35s user 0.11s system 99% cpu 0.467 total

time find ./foo -type f -name "\*.txt" | xargs rm

0.00s user 0.01s system 75% cpu 0.016 total

Clearly using xargs is far more efficient. In fact [several](https://danielmiessler.com/blog/linux-xargs-vs-exec/) [benchmarks](https://www.everythingcli.org/find-exec-vs-find-xargs/) suggest using xargs over exec {} is six times more efficient.