How to use wildcards to match file names in Linux

Searching for files in Linux can seem confusing at first, but don't worry, it gets easier once you understand wildcards. Wildcards are special symbols that help you select multiple files or folders without having to type each name individually.

Searching for files in Linux can seem confusing at first, but don't worry, it gets easier once you understand wildcards. Wildcards are special symbols that help you select multiple files or folders without having to type each name individually. This article will explain how to use wildcards in Linux to efficiently match file names.

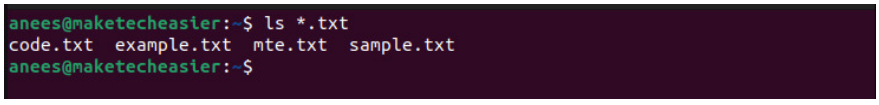
1. File Management in Unix/Linux

1. Asterisk (*)

The asterisk (*) is a Linux wildcard that matches zero or more characters in a file or directory name. It helps to search for, list, or manipulate multiple files at once. It is often used with commands like cp, mv, and rm to perform batch operations.

Match files by extension

We can execute the command **ls *.txt** to match all file names with the .txt extension:



Match files by prefix

If you need to list files that start with the word **example** , you can use the command **ls example*** :

Match files by suffix

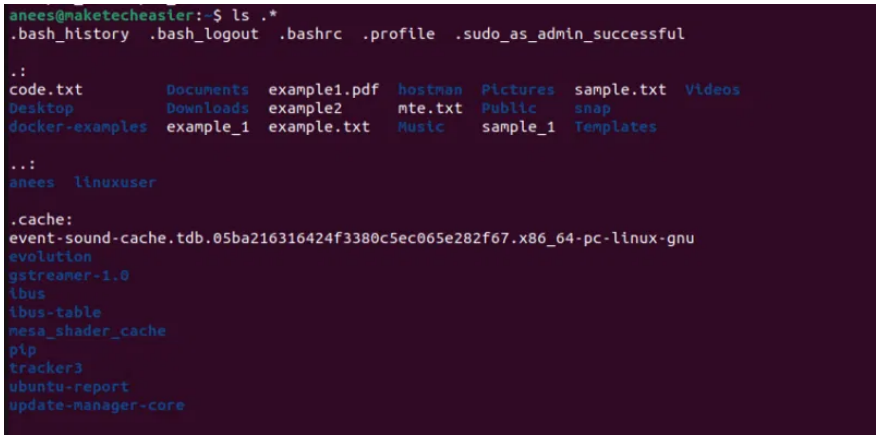
To list or modify files with a certain extension like '_1', use the **ls *_** command :

Match files containing a specific word

We can match file names containing a specific substring by using the asterisk wildcard. For example, the **ls *ample*** command lists all file names containing the substring 'ample':

Match hidden files

In Linux, hidden files start with a dot. We can use the ls .* command to list hidden files:



2. Question mark (?)

The question mark (?) wildcard is used to match a single character in a file name. It helps to find files whose names follow a specific pattern but differ by one character. It is often used to find or manage files with similar names but differ by a single character. For example, **file?.txt** matches "file1.txt", "fileA.txt", "fileB.txt", etc.

Matches files that have any single character at a specific position

We can use the question mark (?) wildcard to match file names where the specified position can be any single character. For example, the command **ls file?.txt** matches any file name that begins with file, followed by any single character, and ends with the .txt extension:

Match files with a fixed number of characters

We can use the question mark wildcard character ? multiple times to match a fixed number of characters in a file name. For example, the command **ls example??.txt** matches any file that starts with the word **example** , followed by any two characters, and ends with the .txt extension:

Combine the wildcard character ? with *

We can combine the ? wildcard with the * wildcard to do some advanced pattern matching. For example, the pattern **?ile*** matches a filename whose first character can be any character, followed by "ile", then any number of characters:

3. Square brackets ([ ])

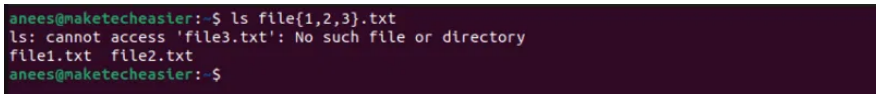
Square brackets ([ ]) match any character enclosed within the brackets. You can include different types of characters, such as letters, numbers, or special symbols, to specify a specific set of matches. For example, the command **ls [1ab]file.txt** lists all files that begin with 1, a, or b, followed by "file.txt":

4. Exclamation mark (!)

We can also negate a set of characters using the ! symbol. For example, the command **ls file[!a-zA-Z]** lists all file names that start with **file** , followed by any character except a letter (az or AZ). It matches "file1", "file_" or "file@" but not "fileA" or "filez":

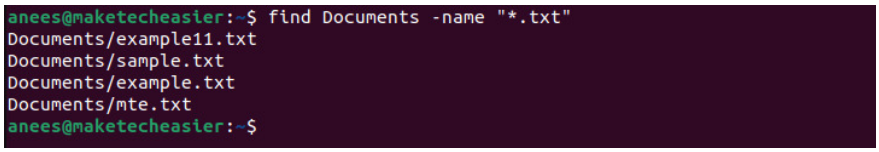
5. Curly brackets ({ })

Curly braces ({ }), also known as range expansions, allow us to specify multiple patterns separated by commas. They expand to specific file names instead of acting as wildcards. For example, the command **ls file{1,2,3}.txt** is equivalent to **ls file1.txt file2.txt file3.txt** . This command lists all of these specific files if they exist:



6. Using wildcards with Linux commands

We can use wildcards with various Linux commands like find , ls , cp and rm to make file management easier by allowing selection based on patterns. For example, we use the command **find Documents -name "*.txt" to locate all .txt files in the Documents** directory :

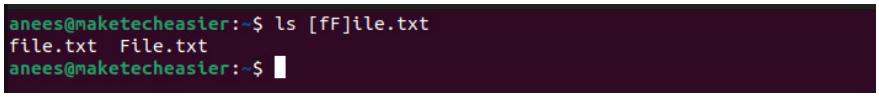


Similarly, we can use wildcards with any other Linux command to achieve a specific purpose.

7. Use wildcards with case-sensitive file names

Wildcards in Linux are case sensitive, meaning file names with different letters are treated as separate. To match both uppercase and lowercase variants, we can use character classes or the case-insensitive option in the command.

For example, we can use the command **ls [fF]ile.txt** to match both file.txt and File.txt:



Now you know how to use wildcards to manage files in Linux faster and easier. Whether you're searching for files, organizing folders, or automating tasks, these wildcard techniques will save you time and effort.

We recommend starting with * and ? as these are the most commonly used. Then experiment with square brackets and curly braces to refine your search. Once you're comfortable, explore regular expressions for more advanced pattern matching.

You finished reading the article "**How to use wildcards to match file names in Linux**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.

© 2019 TipsMake.com