

:mod:`fnmatch` --- Unix filename pattern matching

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\cpython-main [Doc] [library] fnmatch.rst, line 1); [backlink](#)

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\cpython-main [Doc] [library] fnmatch.rst, line 4)

Unknown directive type "module".

```
.. module:: fnmatch
   :synopsis: Unix shell style filename pattern matching.
```

Source code: `:source:`Lib/fnmatch.py``

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\cpython-main [Doc] [library] fnmatch.rst, line 7); [backlink](#)

Unknown interpreted text role "source".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\cpython-main [Doc] [library] fnmatch.rst, line 9)

Unknown directive type "index".

```
.. index:: single: filenames; wildcard expansion
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\cpython-main [Doc] [library] fnmatch.rst, line 11)

Unknown directive type "index".

```
.. index:: module: re
```

This module provides support for Unix shell-style wildcards, which are *not* the same as regular expressions (which are documented in the `:mod:`re`` module). The special characters used in shell-style wildcards are:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\cpython-main [Doc] [library] fnmatch.rst, line 15); [backlink](#)

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\cpython-main [Doc] [library] fnmatch.rst, line 19)

Unknown directive type "index".

```
.. index::
   single: * (asterisk); in glob-style wildcards
   single: ? (question mark); in glob-style wildcards
   single: [] (square brackets); in glob-style wildcards
   single: ! (exclamation); in glob-style wildcards
   single: - (minus); in glob-style wildcards
```

Pattern	Meaning
*	matches everything
?	matches any single character
[seq]	matches any character in <i>seq</i>
[!seq]	matches any character not in <i>seq</i>

For a literal match, wrap the meta-characters in brackets. For example, '[?]' matches the character '?'.
System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-

main\Doc\library\ [cpython-main] [Doc] [library] fnmatch.rst, line 41)

Unknown directive type "index".

```
.. index:: module: glob
```

Note that the filename separator ('/' on Unix) is *not* special to this module. See module `mod:'glob'` for pathname expansion (`mod:'glob'` uses `func:'filter'` to match pathname segments). Similarly, filenames starting with a period are not special for this module, and are matched by the `*` and `?` patterns.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\ [cpython-main] [Doc] [library] fnmatch.rst, line 43); [backlink](#)

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\ [cpython-main] [Doc] [library] fnmatch.rst, line 43); [backlink](#)

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\ [cpython-main] [Doc] [library] fnmatch.rst, line 43); [backlink](#)

Unknown interpreted text role "func".

Also note that `func:'functools.lru_cache'` with the `maxsize` of 32768 is used to cache the compiled regex patterns in the following functions: `func:'fnmatch'`, `func:'fnmatchcase'`, `func:'filter'`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\ [cpython-main] [Doc] [library] fnmatch.rst, line 49); [backlink](#)

Unknown interpreted text role "func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\ [cpython-main] [Doc] [library] fnmatch.rst, line 49); [backlink](#)

Unknown interpreted text role "func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\ [cpython-main] [Doc] [library] fnmatch.rst, line 49); [backlink](#)

Unknown interpreted text role "func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\ [cpython-main] [Doc] [library] fnmatch.rst, line 49); [backlink](#)

Unknown interpreted text role "func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\ [cpython-main] [Doc] [library] fnmatch.rst, line 53)

Unknown directive type "function".

```
.. function:: fnmatch(filename, pattern)
```

```
Test whether the *filename* string matches the *pattern* string, returning
:const:'True' or :const:'False'. Both parameters are case-normalized
using :func:'os.path.normcase'. :func:'fnmatchcase' can be used to perform a
case-sensitive comparison, regardless of whether that's standard for the
operating system.
```

```
This example will print all file names in the current directory with the
extension ``.txt``::
```

```
import fnmatch
import os
```

```
for file in os.listdir('.'):
    if fnmatch.fnmatch(file, '*.txt'):
        print(file)
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\ [cpython-main] [Doc] [library] fnmatch.rst, line 72)

Unknown directive type "function".

```
.. function:: fnmatchcase(filename, pattern)
```

```
Test whether *filename* matches *pattern*, returning :const:`True` or
:const:`False`; the comparison is case-sensitive and does not apply
:func:`os.path.normcase`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\ [cpython-main] [Doc] [library] fnmatch.rst, line 79)

Unknown directive type "function".

```
.. function:: filter(names, pattern)
```

```
Construct a list from those elements of the iterable *names* that match *pattern*. It is the same
``[n for n in names if fnmatch(n, pattern)]``, but implemented more efficiently.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\ [cpython-main] [Doc] [library] fnmatch.rst, line 85)

Unknown directive type "function".

```
.. function:: translate(pattern)
```

```
Return the shell-style *pattern* converted to a regular expression for
using with :func:`re.match`.
```

Example:

```
>>> import fnmatch, re
>>>
>>> regex = fnmatch.translate('*.txt')
>>> regex
'(?s:.*\\.txt)\\Z'
>>> reobj = re.compile(regex)
>>> reobj.match('foobar.txt')
<re.Match object; span=(0, 10), match='foobar.txt'>
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\ [cpython-main] [Doc] [library] fnmatch.rst, line 102)

Unknown directive type "seealso".

```
.. seealso::
```

```
Module :mod:`glob`
    Unix shell-style path expansion.
```