Navigation

- index
- modules |
- next |
- previous |
- Python v2.6.4 documentation »
- The Python Standard Library »
- 11. File and Directory Access »

11.8. `fnmatch` — Unix filename pattern matching¶

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

Pattern	Meaning
`*`	matches everything
`?`	matches any single character
`[seq]`	matches any character in *seq*
`[!seq]`	matches any character not in *seq*

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

`fnmatch.fnmatch`(*filename*, *pattern*)¶

Test whether the *filename* string matches the *pattern* string, returning `True` or `False`. If the operating system is case-insensitive, then both parameters will be normalized to all lower- or upper-case before the comparison is performed. `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
```

`fnmatch.fnmatchcase`(*filename*, *pattern*)¶

Test whether *filename* matches *pattern*, returning `True` or `False`; the comparison is case-sensitive.

`fnmatch.filter`(*names*, *pattern*)¶

Return the subset of the list of *names* that match *pattern*. It is the same as `[n for n in names if fnmatch(n, pattern)]`, but implemented more efficiently.

New in version 2.2.

`fnmatch.translate`(*pattern*)¶

Return the shell-style *pattern* converted to a regular expression.

Example:

```
>>> import fnmatch, re
>>>
>>> regex = fnmatch.translate('*.txt')
>>> regex
'.*\\.txt$'
>>> reobj = re.compile(regex)
>>> reobj.match('foobar.txt')
<_sre.SRE_Match object at 0x...>
```

See also

Module `glob`

Unix shell-style path expansion.

Previous topic

11.7. glob — Unix style pathname pattern expansion

Next topic

11.9. linecache — Random access to text lines

This Page

- Show Source

Navigation

- index
- modules |
- next |
- previous |
- Python v2.6.4 documentation »
- The Python Standard Library »
- 11. File and Directory Access »

Last updated on Feb 26, 2010. Created using Sphinx 0.6.3.