Mercurial > cpython

view Lib/fnmatch.py @ 102611:0de509a79181 2.7


```
Issue #27714: For IDLE's test_textview, backport 3.x subclass with mocks
instead of overriding methods with mocks in original class and module.
This makes the 2.7 test_textview nearly identical to the 3.5/.6 test. [#27714]
```

author Terry Jan Reedy <tjreedy@udel.edu>
date Wed, 10 Aug 2016 19:41:39 -0400 (31 hours ago)
parents fe12c34c39eb
children

line source **line wrap: on**

```
"""Filename matching with shell patterns.

fnmatch(FILENAME, PATTERN) matches according to the local convention.
fnmatchcase(FILENAME, PATTERN) always takes case in account.

The functions operate by translating the pattern into a regular
expression.  They cache the compiled regular expressions for speed.

The function translate(PATTERN) returns a regular expression
corresponding to PATTERN.  (It does not compile it.)
"""

import re

__all__ = ["filter", "fnmatch", "fnmatchcase", "translate"]

_cache = {}
_MAXCACHE = 100

def _purge():
    """Clear the pattern cache"""
    _cache.clear()

def fnmatch(name, pat):
    """Test whether FILENAME matches PATTERN.

    Patterns are Unix shell style:

    *       matches everything
    ?       matches any single character
    [seq]   matches any character in seq
    [!seq]  matches any char not in seq

    An initial period in FILENAME is not special.
    Both FILENAME and PATTERN are first case-normalized
    if the operating system requires it.
    If you don't want this, use fnmatchcase(FILENAME, PATTERN).
    """

    import os
    name = os.path.normcase(name)
    pat = os.path.normcase(pat)
    return fnmatchcase(name, pat)

def filter(names, pat):
```

```
    """Return the subset of the list NAMES that match PAT"""
    import os,posixpath
    result=[]
    pat=os.path.normcase(pat)
    try:
        re_pat = _cache[pat]
    except KeyError:
        res = translate(pat)
        if len(_cache) >= _MAXCACHE:
            _cache.clear()
        _cache[pat] = re_pat = re.compile(res)
    match = re_pat.match
    if os.path is posixpath:
        # normcase on posix is NOP. Optimize it away from the loop.
        for name in names:
            if match(name):
                result.append(name)
    else:
        for name in names:
            if match(os.path.normcase(name)):
                result.append(name)
    return result

def fnmatchcase(name, pat):
    """Test whether FILENAME matches PATTERN, including case.

    This is a version of fnmatch() which doesn't case-normalize
    its arguments.
    """

    try:
        re_pat = _cache[pat]
    except KeyError:
        res = translate(pat)
        if len(_cache) >= _MAXCACHE:
            _cache.clear()
        _cache[pat] = re_pat = re.compile(res)
    return re_pat.match(name) is not None

def translate(pat):
    """Translate a shell PATTERN to a regular expression.

    There is no way to quote meta-characters.
    """

    i, n = 0, len(pat)
    res = ''
    while i < n:
        c = pat[i]
        i = i+1
        if c == '*':
            res = res + '.*'
        elif c == '?':
            res = res + '.'
        elif c == '[':
            j = i
            if j < n and pat[j] == '!':
                j = j+1
            if j < n and pat[j] == ']':
                j = j+1
            while j < n and pat[j] != ']':
                j = j+1
            if j >= n:
```

```
                res = res + '\\['
            else:
                stuff = pat[i:j].replace('\\','\\\\')
                i = j+1
                if stuff[0] == '!':
                    stuff = '^' + stuff[1:]
                elif stuff[0] == '^':
                    stuff = '\\' + stuff
                res = '%s[%s]' % (res, stuff)
        else:
            res = res + re.escape(c)
    return res + '\Z(?ms)'
```