

pathlib ファイル・フォルダ操作

os, os.pathとの区別

- 基本、pathlibを利用していく

```
import pathlib
p = pathlib.Path(r'/Users/mbp441/Desktop') ←または末尾にファイル名
```

処理内容	pathlib	osおよびos.path
カレントディレクトリ取得	p.cwd()	os.getcwd()
先頭の~をホームディレクトリに置換	p.expanduser() p.home()	os.path.expanduser()
パスの存在確認	p.exists()	os.path.exists()
ディレクトリ?	p.is_dir()	os.path.isdir()
ファイル?	p.is_file()	os.path.isfile()
シンボリックか判定	p.is_symlink()	os.path.islink()
絶対パスか判定	p.is_absolute()	os.path.isabs()
絶対パスに変換	p.resolve()	os.path.abspath()
ステータスを取得	p.stat() p.owner() p.group()	os.stat()
パスを連結	p.joinpath()	os.path.join()
ファイル名、フォルダ名を取得	p.name	os.path.basename()
親ディレクトリを取得	p.parent	os.path.dirname()
拡張子を分割・取得	p.suffix	os.path.splitext()
拡張子を除いたファイル名	p.stem	basename = os.path.basename(file_path) file_name = os.path.splitext(basename)[0]

基本

導入

```
import pathlib  
p = pathlib.Path(r'/Users/mbp441/Desktop')
```

カレント取得

```
p.cwd()  
# >> PosixPath('/Users/mbp441/Desktop/github/PYTHON/jupyter_notebook')
```

ホームディレクトリ取得

```
p.home()  
# >> PosixPath('/Users/mbp441')
```

絶対パスを取得

```
p.resolve()  
# >> PosixPath('/Users/mbp441/Desktop')
```

```
import pathlib  
p = pathlib.Path('/Users/mbp441/Desktop/test.csv')
```

以下、いずれも()は不要

親ディレクトリを取得

```
p.parent  
# >> PosixPath('/Users/mbp441/Desktop')
```

ファイル名を取得

```
p.name  
# >> 'test.csv'
```

拡張子を取得

```
p.suffix  
# >> '.csv'
```

拡張子を除いたファイル名を取得

```
p.stem  
# >> 'test'
```

フォルダ作成

同名フォルダがない場合

```
p = pathlib.Path('/Users/mbp441/Desktop/fol_1')
p.mkdir()
```

同名フォルダがある場合

- 同名フォルダがあるとエラーになる(`FileExistsError: ~`)
- 「`exist_ok=True`」
 - 同名フォルダがあってもエラーを表示しない
 - また、**上書きもしない**
- cf.「`parents=True`」: 親ディレクトリが無ければ作成する

```
p = pathlib.Path('/Users/mbp441/Desktop/fol_1')
p.mkdir(exist_ok=True)
```

空ファイルの作成

```
p = pathlib.Path('/Users/mbp441/Desktop/test.csv')
p.touch()
# ファイルが既存の場合は、中身はそのまま、タイムスタンプだけが更新される

# ファイルが既存の場合はエラーを出す
p.touch(exist_ok=False)

? xlsxとかのファイルは作成できる? 壊れる?
```

ファイル・フォルダ名の変更

- フォルダ名も変更可能

```
p = pathlib.Path(base_dir)
before_f = p / 'before.txt'
after_f = p / 'after.txt'
before_f.rename(pathlib.Path(after_f))
```

同名ファイルがあれば～

```
p = pathlib.Path("/Users/mbp441/Desktop/aaa.csv")

if p.is_file():
    while True:
        answer = input('同名ファイルあり。上書きしますか？ (y or n):')
        if answer == 'y':
            df.to_csv("/Users/mbp441/Desktop/aaa.csv")
            print('上書きしました。')
            break
        elif answer == 'n':
            print('処理を中断します。')
            break
    else:
        print('ファイルを新規作成します。')
        df.to_csv("/Users/mbp441/Desktop/aaa.csv")
```

```
cf. p.is_dir(), p.exists()
```

削除

```
p = pathlib.Path('/Users/mbp441/Desktop/test.csv')
```

ファイルを削除

```
p.unlink()
```

該当ファイルがない場合はエラーになる。エラーを非表示にする場合は以下

```
p.unlink(missing_ok=True)
```

フォルダ(空)を削除

```
p = pathlib.Path('/Users/mbp441/Desktop/fol_1')
p.rmdir()
```

cf. rmdirは中身が空フォルダしか削除不可

またpathlib.Pathには中身ごと削除するメソッドはない

フォルダ(中身あり)を削除 **※使用注意**

```
import pathlib
import shutil
```

```
p = pathlib.Path('/Users/mbp441/Desktop/fol_1')
shutil.rmtree(p)
```

一覧表示・抽出

フォルダ内の一覧表示

```
p = pathlib.Path('/Users/mbp441/Desktop/fol_1')

for f in p.iterdir():
    print(f)

f_list = [f for f in p.iterdir()]
```

ファイルのみを表示

```
for f in p.iterdir():
    if f.is_file():
        print(x)

f_list = [f for f in p.iterdir() if f.is_file()]
```

フォルダのみを表示

```
for f in p.iterdir():
    if f.is_dir():
        print(f)

dir_list = [f for f in p.iterdir() if f.is_dir()]
```

フォルダから特定ファイルだけを抽出したい

```
p = pathlib.Path('/Users/mbp441/Desktop/fol_1')

# フォルダ直下のファイルを取り出す場合
list_csv = list( p.glob('*.csv') )

# サブフォルダ含めたファイルを取り出す場合(再帰的)
list_csv = list( p.glob('**/*.csv') )
```

※場合によりstr化する必要あり

Shutilによるファイル・フォルダのコピー

```
import shutil
```

```
src_f    = pathlib.Path("/Users/mbp441/Desktop/aaa.csv")
dst_f    = pathlib.Path("/Users/mbp441/Desktop/bbb.csv")

src_dir  = pathlib.Path("/Users/mbp441/Desktop/test_fol1")
dst_dir  = pathlib.Path("/Users/mbp441/Desktop/test_fol2")
```

ファイルのコピー

```
shutil.copy(src_f, dst_f)
```

ファイルをフォルダー内へコピー

```
shutil.copy(src_f, dst_dir)
```

フォルダをコピー(dst_dirが**未作成**の場合)

```
shutil.copytree(src_dir, dst_dir)
```

拡張子が「*.txt」**以外**のものをコピー(〃)

```
shutil.copytree(src_dir, dst_dir, ignore=shutil.ignore_pattern('*.txt'))
```

フォルダをコピー(dst_dirが**すでに存在する場合**はエラーになるため)

→？上書きはできないはず。一度削除してからコピーするなどするしかない？

→たぶん以下でいいはず。

```
if not p.is_dir(): ← p.exists()でもよいは
    p.mkdir()
else:
    shutil.rmtree(p)
    p.mkdir()
```