







俺はpathlibで行く！

はんなりPython #32 標準ライブラリなLT会

2020/09/13 Hiroshi Sano

お前誰よ

- 佐野浩士 (Hiroshi Sano) [@hrs_sano645](#)
- 静岡県の富士市 
- Job 
 - 佐野設計事務所    設計以外何でも屋
 - 米農家 
- Community 
 -  : [shizuoka.py](#), [unagi.py](#), Python駿河
 - : PyCon mini Shizuokaスタッフ
 - : PyCon JP 2020 チュートリアル講師

 PyCon JP 2020 チュートリアルお疲れ様でした 

- はんなりPythonの会
- PyData Osaka
- PyCon mini Hiroshima

推しライブラリ

pathlib

<https://docs.python.org/ja/3/library/pathlib.html>

pathlib知ってる？

- 知ってる人、カメラorZoomの機能で挙手してください👋

os.pathの置き換え

os.pathは昔からある（1系から）

<https://docs.python.org/ja/3/library/os.path.html>

昔からお世話になっていたけど、不満点

不満点

とくに区切り文字！

- Winだと `\\` , Unixだと `/`
- ハードコードすると結構トラブル起こしがち

os.pathと書いて操作するとコードが汚い

区切り文字混ぜるな危険

```
# Ubuntu 18.04です
>>> import os.path
>>> os.path.abspath("Documents\\test")
'/home/hiroshi/Documents/Documents\\test'
```

os.pathと毎回書くと面倒

```
>>> os.path.join(os.path.abspath("Document"), "test")  
'/home/hiroshi/Documents/Document/test'
```

エイリアスでしのぐしか...

そこでpathlib

- Python3.4から使える
- Pathオブジェクトとして操作 -> 統一性がある
- os, os.path, globといったパスを扱うモジュールをこれ1つで対応できる

pathlibのいいところ

オブジェクトとして管理

```
# Windowsでやってる
>>> from pathlib import Path
>>> p = Path("~/Documents/hellopathlib.txt")
>>> p
WindowsPath('~\\Documents\\hellopathlib.txt')
```

パス区切り文字をどちらでもいい

```
# winの区切り文字も対応
>>> win_kugiri = Path("~\\Documents\\hellopathlib.txt")
>>> win_kugiri
WindowsPath('~\\Documents\\hellopathlib.txt')

>>> posix_kugiri = Path("~/Documents/hellopathlib.txt")
>>> posix_kugiri
WindowsPath('~\\Documents\\hellopathlib.txt')
```

APIに統一感がある

os.pathはパスを扱う。ファイルアクセスは別モジュール

- os.mkdirとかありますよね -> osモジュール
- 検索のときにglobモジュール使いますよね
- 組み込みのopen関数

pathlibはファイル操作もできる

- Path.mkdirができる
- globもできる -> Path.glob, Path.rglob（再帰検索専用）
- Path.openメソッドが使える（openとほぼ変わりはないです）

osモジュールとの対応表

os モジュールにあるツールとの対応付け

スラッシュ（除算演算子）をパスのjoinにつかえる

これがとてもいい

```
>>> from pathlib import Path
>>> Path.home()
WindowsPath('C:/Users/hiroshi')
>>> Path.home() / "Documents"
WindowsPath('C:/Users/hiroshi/Documents')
>>> home_docsdir = Path.home() / "Documents"
>>> home_docsdir.is_dir()
True
```

パスにまつわる操作系はほぼ全部pathlibからできる

俺はpathlibで行く！ はんなりPython 32 標準ライブラリなLT会

注意点

古めのライブラリだとPathオブジェクトを扱えない

(文字列としてパスをとる場合が大体)

str関数に通すと文字列にできる

```
# os.pathは文字列を生成する
>>> type(os.path.abspath("\\Documents"))
<class 'str'>
# pathlibはPathオブジェクト
>>> type(home_docs)
<class 'pathlib.WindowsPath'>
# str関数に渡すと文字列
>>> str(home_docs)
'C:\\Users\\hiroshi\\Documents'
>>> type(str(home_docs))
<class 'str'>
>>>
```

ライブラリ側で対応したい

os.PathLikeオブジェクトへ対応させる

(3.6から追加された特殊メソッドへの対応)

<https://docs.python.org/ja/3/library/os.html#os.PathLike>

もしくは3.6以下の互換を捨てて内部でpathlibのオブジェクトを使ってしまう

俺はpathlibで行く！ はんなりPython 32 標準ライブラリなLT会

まとめ

pathlibは操作の統一性があるので悩み減ります

Python3.4以降はpathlibを使いましょう💪