Pathlib

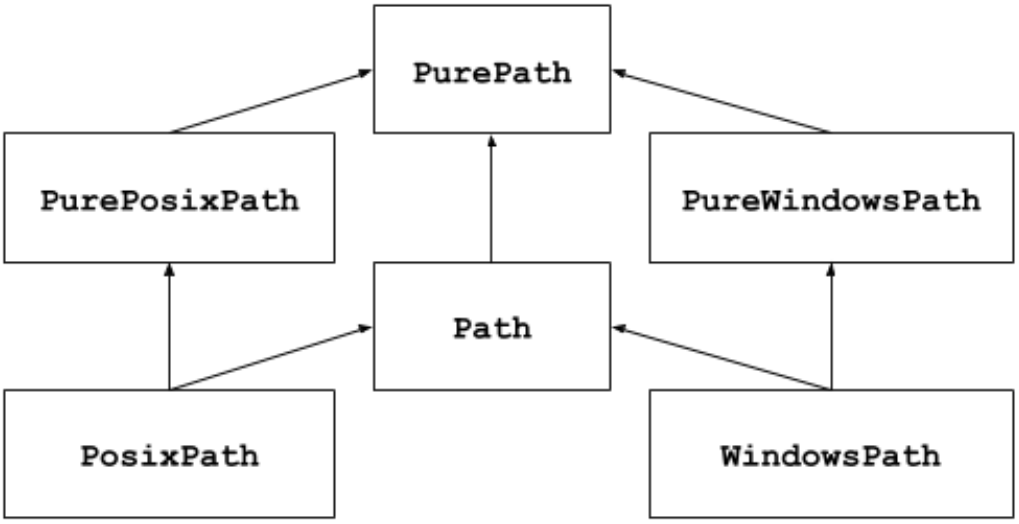
September 17, 2024

1 Pathlib

Pathlib is an object oriented module for handling paths. It is part of the standard library since Python 3.4.

https://docs.python.org/3/library/pathlib.html

While it has special classes for handling Windows and Posix paths, the most common use is through the *Path* class.



Please note that under Linux, you will get PosixPath objects. On Windows, you will get WindowsPath objects.

The *Pure* superclasses implement functionality that doesn't access the file system, like path calculations.

1.0.1 Iteration over directory contents

[5]:
```
from pathlib import Path

p = Path('.')
[x for x in p.iterdir() if x.is_dir()]
```

[5]:
```
[PosixPath('.ipynb_checkpoints'),
 PosixPath('.pytest_cache'),
 PosixPath('__pycache__')]
```

1.0.2 Globbing

[6]:
```
list(p.glob('**/*.py'))
```

[6]:
```
[PosixPath('nested.py'),
 PosixPath('tested_code.py'),
 PosixPath('untitled.py'),
 PosixPath('.ipynb_checkpoints/nested-checkpoint.py'),
 PosixPath('.ipynb_checkpoints/tested_code-checkpoint.py'),
 PosixPath('.ipynb_checkpoints/untitled-checkpoint.py')]
```

1.0.3 Path concatenation

Initial object needs to be a *Path*, following objects can be *Path* or *str*.

[14]:
```
p = Path('/usr')
q = p / 'bin' / 'pyversions'  # / is a path concatenator operator
q
```

[14]:
```
PosixPath('/usr/bin/pyversions')
```

1.0.4 Following links

[15]:
```
q.resolve()  # Follow soft links
```

[15]:
```
PosixPath('/usr/share/python/pyversions.py')
```

1.0.5 Checking for existence

We can build paths without accessing the file system. We can then check if the path refers to an actual object in the file system.

[16]:
```
q.exists()
```

[16]:
```
True
```

1.0.6 Checking properties of what the path refers to

[17]:
```
q.is_dir()
```

[17]:
```
False
```