제9강 Organizing Files

학습 목차

- 파일과 폴더의 복사 이동 삭제
- 휴지통 액세스
- ZIP 압축과 해제
- 파일 정보
- 파일 비교

파일과 폴더 복사와 이동

shutil.copy(src, dst)

shutil.copytree(src, dst)

shutil.move(src, dst)

파일과 폴더 삭제

os.unlink(path)

os.remove(path)

os.rmdir(path)

shutil.rmtree(path)

삭제는 항상 주의 필요!

```
import os
for filename in os.listdir():
    if filename.endswith('.rxt'):
        #os.unlink(filename)
        print(filename)
```

send2trash – 휴지통 보내기

```
>>> import send2trash
>>> baconFile = open('bacon.txt', 'a')   # creates the file
>>> baconFile.write('Bacon is not a vegetable.')
25
>>> baconFile.close()
>>> send2trash.send2trash('bacon.txt')
```

휴지통 액세스 – 윈도우 winshell 모듈

```
import winshell

r = list(winshell.recycle_bin())
for index,value in enumerate(r):
      print(index,value.original_filename())

index = r.index("C:\ORIGINAL\PATH\test.txt")
winshell.undelete(r[index].original_filename())
```

Zip 파일 읽기

```
>>> import zipfile, os
>>> os.chdir('C:\\')    # move to the folder with example.zip
>>> exampleZip = zipfile.ZipFile('example.zip')
>>> exampleZip.namelist()
['spam.txt', 'cats/', 'cats/catnames.txt', 'cats/zophie.jpg']
>>> spamInfo = exampleZip.getinfo('spam.txt')
>>> spamInfo.file_size
13908
>>> spamInfo.compress_size
3828
>>> 'Compressed file is %sx smaller!' % (round(spamInfo.file_size / spamInfo
.compress_size, 2))
'Compressed file is 3.63x smaller!'
>>> exampleZip.close()
```

Zip 파일 추출

```
>>> exampleZip = zipfile.ZipFile('example.zip')
>>> exampleZip.extractall()
>>> exampleZip.close()
```

```
>>> exampleZip.extract('spam.txt')
'C:\\spam.txt'
>>> exampleZip.extract('spam.txt', 'C:\\some\\new\\folders')
'C:\\some\\new\\folders\\spam.txt'
>>> exampleZip.close()
```

Zip 파일 만들고 추가하기

```
>>> import zipfile
>>> newZip = zipfile.ZipFile('new.zip', 'w')
>>> newZip.write('spam.txt', compress_type=zipfile.ZIP_DEFLATED)
>>> newZip.close()
```

파일 정보

os.path.getsize()

os.path.getmtime() – 변경 시각

os.path.getctime() – 생성 시각

파일과 폴더 비교

filecmp.cmp(f1, f2, shallow=True)

filecmp.cmpfiles(dir1, dir2, common, shallow=True)

```
from filecmp import dircmp
def print_diff_files(dcmp):
    for name in dcmp.diff_files:
        print("diff_file %s found in %s and %s" % (name, dcmp.left,
              dcmp.right))
    for sub_dcmp in dcmp.subdirs.values():
        print_diff_files(sub_dcmp)

dcmp = dircmp('dir1', 'dir2')
print_diff_files(dcmp)
```