4.1 檔案和目錄管理 / File and directory management // Manajemen file dan direktori

4.1.1 os

os 提供建立目錄、刪除目錄、刪除檔案、執行作業系統命令等方法，使用時必須匯入os 套件。

/ os provides methods for creating directories, deleting directories, deleting files, executing operating system commands, etc., and must be imported into the os suite.

// os menyediakan metode untuk membuat direktori, menghapus direktori, menghapus file, menjalankan perintah sistem operasi, dll., dan harus diimpor ke dalam os suite.

remove() 方法 /remove() method //remove() metode

刪除指定的檔案，一般都會配合 os.path 的 exists() 方法，先檢查該檔案是否存在，再決定是否要刪除檔案。

/ Deleting the specified file will generally match the exists() method of os.path, first check if the file exists, and then decide whether to delete the file.

// Menghapus file yang ditentukan biasanya akan cocok dengan metode exist () os.path, pertama periksa apakah file tersebut ada, dan kemudian putuskan apakah akan menghapus file tersebut.

In [ ]:

```
import os

file = "myFile.txt"

if os.path.exists(file):
    os.remove(file)
else:
    print(file + "檔案未建立!")
```

mkdir() 方法 /mkdir() method //mkdir() metode

利用 mkdir() 方法可以建立指定的目錄。

/ The specified directory can be created using the mkdir() method.

// Direktori yang ditentukan dapat dibuat menggunakan metode mkdir ().

In [ ]:

```
import os

dir = "myDir"

if not os.path.exists(dir):
    os.mkdir(dir)
else:
    print(dir + "已經建立!")
```

rmdir() 方法 / rmdir() method // rmdir() metode

rmdir() 方法可以刪除指定的目錄，刪除目錄前必須先刪除該目錄的檔案。一般都會先檢查該目錄是否已經建立，再決定是否要刪除目錄。

/ The rmdir() method can delete the specified directory. Before deleting the directory, you must delete the file of the directory. Generally, it is checked whether the directory has been created before deciding whether to delete the directory.

// Metode rmdir () dapat menghapus direktori yang ditentukan. Sebelum menghapus direktori, Anda harus menghapus file direktori. Umumnya, ini diperiksa apakah direktori telah dibuat sebelum memutuskan apakah akan menghapus direktori.

In [ ]:

```
import os

dir = "myDir"

if os.path.exists(dir):
    os.rmdir(dir)
else:
    print(dir + "目錄未建立!")
```

system() 方法 / system() method // system() metode

執行作業系統命令。

/ Execute the operating system command.

// Jalankan perintah sistem operasi.

例如：清除螢幕、建立 dir2 目錄、複製 <ossystem.py> 到 dir2 目錄中，檔名為 <copyfile.py>， 最後以記事本開啟 <copyfile.py> 檔。

/ For example: clear the screen, create the dir2 directory, copy <ossystem.py> to the dir2 directory, the file name is <copyfile.py>, and finally open the <copyfile.py> file with Notepad.

// Sebagai contoh: bersihkan layar, buat direktori dir2, salin <ossystem.py> ke direktori dir2, nama file adalah <copyfile.py>, dan akhirnya buka file <copyfile.py> dengan Notepad. </copyfile.py></ossystem.py></copyfile.py></copyfile.py></ossystem.py></copyfile.py></copyfile.py></ossystem.py>

In [ ]:

```
import os

cur_path = os.path.dirname(__file__) # 取得目前路徑
os.system("cls")  # 清除螢幕
os.system("mkdir dir2")  # 建立 dir2 目錄
os.system("copy ossystem.py dir2\copyfile.py") # 複製檔案
file = cur_path + "\dir2\copyfile.py"
os.system("notepad " + file)  # 以記事本開啟 copyfile.py 檔
```

4.1.2 os.path

os.path 用以處理檔案路徑和名稱，檢查檔案或路徑是否存在，也可以計算檔案的大小。首先必須匯入 os.path 套件

/ os.path is used to process the file path and name, check the existence of the file or path, and calculate the size of the file. First you must import the os.path package

// os.path digunakan untuk memproses path dan nama file, memeriksa keberadaan file atau path, dan menghitung ukuran file. Pertama Anda harus mengimpor paket os.path

os.path 提供下列的方法：

/ os.path provides the following methods:

// os.path menyediakan metode berikut:

方法 / Method // Metode	**說明 / Description // Deskripsi**

os.path.abspath(path)	/ Return a normalized absolutized version of the pathname path. // Kembalikan versi versi path path path yang dinormalkan dan dinormalisasi.
os.path.basename(path)	/ Return the base name of pathname path. // Kembalikan nama dasar jalur path nama.
os.path.dirname(path)	/ Return the directory name of pathname path. // Kembalikan nama direktori path path nama.
os.path.exists(path)	/ Return True if path refers to an existing path or an open file descriptor. Returns False for broken symbolic links. // Return True jika path mengacu pada path yang ada atau deskriptor file terbuka. Mengembalikan Salah untuk tautan simbolis yang rusak.
os.path.getsize(path)	/ Return the size, in bytes, of path. Raise OSError if the file does not exist or is inaccessible. // Kembalikan ukuran, dalam byte, jalur. Naikkan OSError jika file tidak ada atau tidak dapat diakses.
os.path.isabs(path)	/ Return True if path is an absolute pathname. On Unix, that means it begins with a slash, on Windows that it begins with a (back)slash after chopping off a potential drive letter. // Return True jika path adalah nama path absolut. Pada Unix, itu berarti dimulai dengan garis miring, pada Windows yang dimulai dengan slash (kembali) setelah memotong huruf drive potensial.
os.path.isfile(path)	/ Return True if path is an existing regular file. This follows symbolic links, so both islink() and isfile() can be true for the same path. // Return True jika path adalah file reguler yang ada. Ini mengikuti tautan simbolis, sehingga baik islink () dan isfile () dapat benar untuk jalur yang sama.
os.path.isdir(path)	/ Return True if path is an existing directory. This follows symbolic links, so both islink() and isdir() can be true for the same path. // Return True jika path adalah direktori yang ada. Ini mengikuti tautan simbolis, sehingga baik islink () dan isdir () bisa benar untuk jalur yang sama.
os.path.split(path)	/ Split the pathname path into a pair, (head, tail) where tail is the last pathname component and head is everything leading up to that. The tail part will never contain a slash; if path ends in a slash, tail will be empty. If there is no slash in path, head will be empty. If path is empty, both head and tail are empty. Trailing slashes are stripped from head unless it is the root (one or more slashes only). In all cases, join(head, tail) returns a path to the same location as path (but the strings may differ). // Pisahkan path path nama menjadi sepasang, (kepala, ekor) di mana ekor adalah komponen pathname terakhir dan kepala adalah segalanya yang mengarah ke sana. Bagian ekor tidak akan pernah mengandung garis miring; jika jalur berakhir dengan garis miring, ekor akan kosong. Jika tidak ada garis miring di jalan, kepala akan kosong. Jika jalan kosong, kepala dan ekor kosong. Trailing slash dilucuti dari kepala kecuali itu adalah root (satu atau lebih tebasan saja). Dalam semua kasus, bergabung (kepala, ekor) mengembalikan jalur ke lokasi yang sama sebagai jalur (tetapi string mungkin berbeda).
	/ Split the pathname path into a pair (drive, tail) where drive is either a mount point or the empty string. On systems which do not use drive specifications, drive will always be the empty string. In all cases, drive + tail will be the same

os.path.splitdrive(path)	as path.On Windows, splits a pathname into drive/UNC sharepoint and relative path.If the path contains a drive letter, drive will contain everything up to and including the colon. e.g. splitdrive("c:/dir") returns ("c:", "/dir")If the path contains a UNC path, drive will contain the host name and share, up to but not including the fourth separator. e.g. splitdrive("//host/computer/dir") returns ("//host/computer", "/dir") // Pisahkan path path nama menjadi sepasang (drive, ekor) di mana drive adalah titik mount atau string kosong. Pada sistem yang tidak menggunakan spesifikasi drive, drive akan selalu menjadi string kosong. Dalam semua kasus, drive + tail akan sama dengan path.,Pada Windows, membagi pathname menjadi drive / UNC sharepoint dan path relatif.,Jika path berisi huruf drive, drive akan berisi semuanya hingga dan termasuk titik dua. misalnya splitdrive ("c: / dir") mengembalikan ("c:", "/ dir"),Jika path berisi jalur UNC, drive akan berisi nama host dan share, hingga tetapi tidak termasuk pemisah keempat. misalnya splitdrive ("// host / komputer / dir") mengembalikan ("// host / komputer", "/ dir")
os.path.join(path, *paths)	/ Join one or more path components intelligently. The return value is the concatenation of path and any members of *paths with exactly one directory separator (os.sep) following each non-empty part except the last, meaning that the result will only end in a separator if the last part is empty. If a component is an absolute path, all previous components are thrown away and joining continues from the absolute path component.On Windows, the drive letter is not reset when an absolute path component (e.g., r'\foo') is encountered. If a component contains a drive letter, all previous components are thrown away and the drive letter is reset. Note that since there is a current directory for each drive,os.path.join("c:", "foo") represents a path relative to the current directory on drive C:(c:foo), not c:\foo. // Bergabunglah dengan satu atau lebih komponen jalan secara cerdas. Nilai kembalian adalah penggabungan jalur dan setiap anggota* jalur dengan tepat satu pemisah direktori (os.sep) mengikuti setiap bagian yang tidak kosong kecuali yang terakhir, yang berarti bahwa hasilnya hanya akan berakhir dengan separator jika bagian terakhir kosong . Jika komponen adalah jalur absolut, semua komponen sebelumnya dibuang dan bergabung berlanjut dari komponen jalur absolut.,Pada Windows, huruf kandar tidak diatur ulang ketika komponen path absolut (misalnya, r '\ foo') ditemukan. Jika komponen berisi huruf kandar, semua komponen sebelumnya dibuang dan huruf kandar disetel ulang. Perhatikan bahwa karena ada direktori saat ini untuk setiap drive, os.path.join ("c:", "foo") mewakili jalur relatif ke direktori saat ini di drive C: (c: foo), bukan c: \ foo .

例如：取得目前路徑、完整路徑名稱、檔案大小、最後的檔案或路徑名稱、偵測是否為目錄、將路徑分解為路徑和檔名、取得磁碟機名稱等。

/ For example: get the current path, full path name, file size, last file or path name, detect whether it is a directory, break the path into path and file name, get the name of the drive, and so on.

// Sebagai contoh: dapatkan jalur saat ini, nama path lengkap, ukuran file, file terakhir atau nama path, mendeteksi apakah itu direktori, memutuskan path ke path dan nama file, mendapatkan nama drive, dan seterusnya.

In [ ]:

```
import os.path

cur_path = os.path.dirname(__file__) # 取得目前目錄路徑

print("現在目錄路徑："+cur_path)

filename=os.path.abspath("ospath.py")

if os.path.exists(filename):
    print("完整路徑名稱：" + filename)
    print("檔案大小：" , os.path.getsize)

    basename=os.path.basename(filename)
    print("最後的檔案或路徑名稱：" + basename)

    dirname=os.path.dirname(filename)
    print("目前檔案目錄路徑：" + dirname)

    print("是否為目錄：",os.path.isdir(filename))

    fullpath,fname=os.path.split(filename)
    print("目錄路徑：" + fullpath)
    print("檔名：" + fname)

    Drive,fpath=os.path.splitdrive(filename)
    print("磁碟機：" + Drive)
    print("路徑名稱：" + fpath)

    fullpath = os.path.join(fullpath + "\\" + fname)
    print("組合路徑= " + fullpath)
```