

碁峯資訊

版權聲明：本教學投影片僅供教師授課講解使用，投

```
import pygame, random, time

class Ball(pygame.sprite.Sprite):
    dx = 0
    dy = 0
    x = 0
    y = 0
    direction = 0
    speed = 0

    def __init__(self, sp, srx, sry, radium, color):
        pygame.sprite.Sprite.__init__(self)
        self.speed = sp
        self.x = srx
        self.y = sry
        self.image = pygame.Surface([radium*2, radium*2])
        self.image.fill((255,255,255))
        pygame.draw.circle(self.image, color, (radium,radium), radium, 0)
        self.rect = self.image.get_rect()
        self.rect.center = (srx,sry)
        self.direction = random.randint(40,70)

    def update(self):
        radian = math.radians(self.direction)
        self.dx = self.speed * math.cos(radian)
        self.dy = -self.speed * math.sin(radian)
        self.x += self.dx
        self.y += self.dy
        self.rect.x = self.x
        self.rect.y = self.y
        if(self.rect.left <= 0 or self.rect.right >= screen.get_width()-10):
            self.bouncelr()
        elif(self.rect.top <= 0 or self.rect.bottom >= screen.get_height()-10):
            self.bounceup()
        else:
            return True
        return False

    def bouncelr(self):
        self.direction = 360 - self.direction
        self.direction = (180 - self.direction) % 360

class Brick(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)
        self.image = pygame.image.load("media\\brick.png")
        self.image.convert()
        self.rect = self.image.get_rect()
        self.rect.x = int((screen.get_width() - self.rect.width)/2)
        self.rect.y = screen.get_height() - self.rect.height - 20

class Pad(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)
        self.image = pygame.image.load("media\\pad.png")
        self.image.convert()
        self.rect = self.image.get_rect()
        self.rect.x = int((screen.get_width() - self.rect.width)/2)
        self.rect.y = screen.get_height() - self.rect.height - 20

def update(self):
    pos = pygame.mouse.get_pos()
    self.rect.x = pos[0]

def gameover(message):
    global running
    text = font1.render(message, 1, (255,0,255))
    screen.blit(text, (screen.get_width()/2-100,screen.get_height()/2-20))
    pygame.display.update()
    time.sleep(3)
    running = False

pygame.init()
score = 0
font = pygame.font.SysFont("SimHei", 20)
font1 = pygame.font.SysFont("SimHei", 32)
soundhit = pygame.mixer.Sound("media\\hit.wav")
soundpad = pygame.mixer.Sound("media\\pad.wav")
```

Python

Python Beginner Course

初學特訓班

增訂版

04

Python 能夠大量快速的處理電腦系統中的檔案與資料夾，除了使用 `os` 套件進行目錄建立與刪除目錄、檔案刪除、執行作業系統命令等動作，也可以利用Python內建的`open()` 函式開啟指定的檔案，並進行檔案內容的讀取、寫入或修改。

Python 內建嵌入式資料庫SQLite，利用檔案儲存整個資料庫，SQLite 的特點是可以使用 SQL 語法管理資料庫，執行新增、修改、刪除和查詢。

檔案處理與SQLite資料庫

4.1 檔案和目錄管理

4.2 File 檔案

4.3 SQLite 資料庫

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。(<osremove.py>)

/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.

```
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 ().

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

() 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。(<osrmdir.py>)

/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.

```
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。（<ossystem.py>）

/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.

```
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	說明
<code>abspath()</code>	傳回檔案完整的路徑名稱
<code>basename()</code>	傳回檔案路徑名稱最後的檔案或路徑名稱。如果測試的是檔案會傳回檔名，測試的是路徑會傳回路徑。
<code>dirname()</code>	傳回指定檔案完整的目錄路徑， <code>dirname(__file__)</code> 則可以取得目前的目錄路徑。
<code>exists()</code>	檢查指定的檔案或路徑是否存在
<code>getsize</code> 屬性	取得指定檔案的大小 (Bytes)
<code>isabs()</code>	檢查指定路徑是否為完整路徑名稱
<code>isfile()</code>	檢查指定路徑是否為檔案
<code>isdir()</code>	檢查指定路徑是否為目錄
<code>split()</code>	分割檔案路徑名稱為目錄路徑和檔案
<code>splitdrive()</code>	分割檔案路徑名稱為磁碟機和檔案路徑名稱
<code>join()</code>	將路徑和檔案名稱結合為完整路徑

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。(<ospath.py>)

/ 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.

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。(<ospath.py>)

```
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)

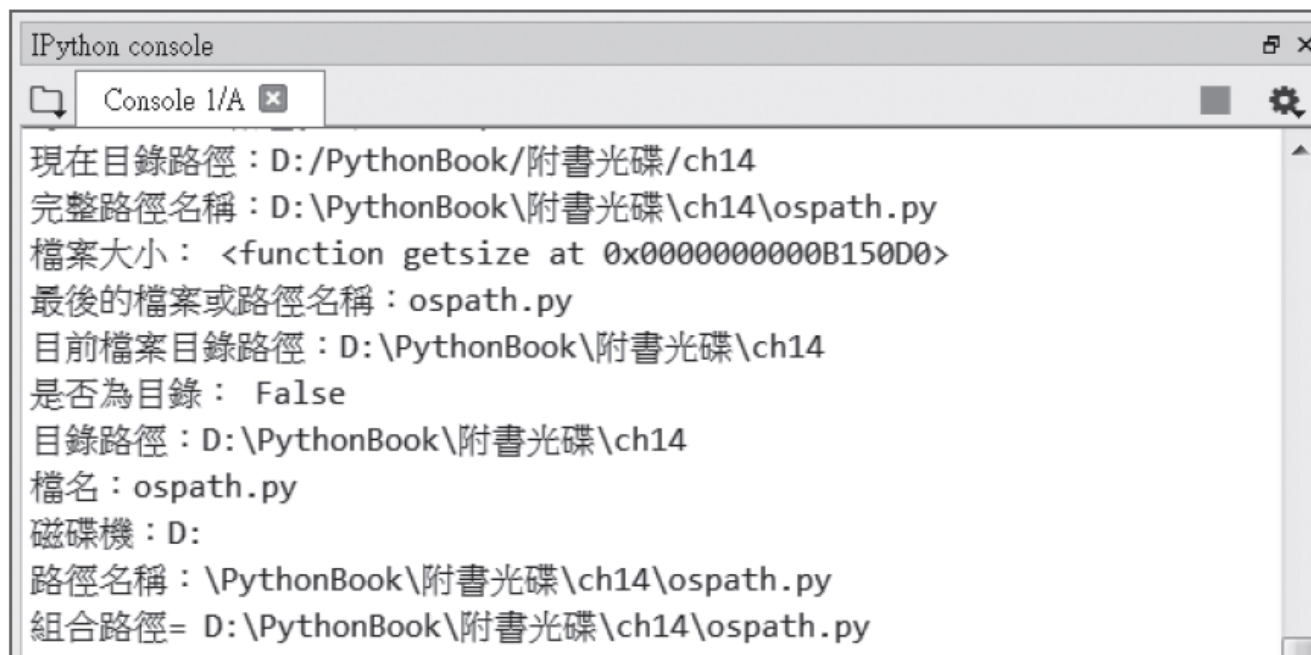
    Drive,fpath=os.path.splitdrive(filename)
    print("磁碟機：" + Drive)
    print("路徑名稱：" + fpath)

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

使用 IPython console 執行。

/Execute using the IPython console.

//Jalankan menggunakan konsol IPython.



```
IPython console
Console 1/A
現在目錄路徑：D:/PythonBook/附書光碟/ch14
完整路徑名稱：D:\PythonBook\附書光碟\ch14\ospath.py
檔案大小： <function getsize at 0x0000000000B150D0>
最後的檔案或路徑名稱：ospath.py
目前檔案目錄路徑：D:\PythonBook\附書光碟\ch14
是否為目錄： False
目錄路徑：D:\PythonBook\附書光碟\ch14
檔名：ospath.py
磁碟機：D:
路徑名稱：\PythonBook\附書光碟\ch14\ospath.py
組合路徑= D:\PythonBook\附書光碟\ch14\ospath.py
```

4.1.3 os.walk

為了方便說明，本範例檔刻意放在本章的 <oswalk> 目錄下，該目錄包含了 <Dir> 目錄和 <oswalk.py>、<oswalk1.txt> 檔，並在 <Dir> 目錄下又建立了 <SubDir> 目錄和 <Dir1.txt>、<Dir2.txt> 檔，同時在 <SubDir> 目錄也建立了檔案 <SubDir1.txt> 檔。架構如下：(<oswalk.py>)

/For convenience of explanation, this sample file is deliberately placed in the <oswalk> directory of this chapter. This directory contains the <Dir> directory and the <oswalk.py>, <oswalk1.txt> files, and is created in the <Dir> directory. <SubDir> directory and <Dir1.txt>, <Dir2.txt> files, and file <SubDir1.txt> file is also created in the <SubDir> directory. The architecture is as follows:

//Untuk memudahkan penjelasan, file contoh ini sengaja ditempatkan di direktori <oswalk> pada bab ini. Direktori ini berisi direktori <Dir> dan file <oswalk.py>, <oswalk1.txt>, dan dibuat di direktori <Dir>. File <SubDir> dan file <Dir1.txt>, <Dir2.txt>, dan file <SubDir1.txt> juga dibuat di direktori <SubDir>. Arsitekturnya adalah sebagai berikut:

```
\oswalk
├── \Dir ─────────── \SubDir ───────── SubDir1.txt
├── oswalk.py        ─── Dir1.txt
└── oswalk1.txt      ─── Dir2.txt
```

```
import os
cur_path=os.path.dirname(__file__) # 取得目前路徑
sample_tree=os.walk(cur_path)
for dirname,subdir,files in sample_tree:
    print(" 檔案路徑：",dirname)
    print(" 目錄串列：", subdir)
    print(" 檔案串列：",files)
    print()
```

使用 IPython console 執行。

/Execute using the IPython console.

//Jalankan menggunakan konsol IPython.



The screenshot shows a window titled "IPython console" with a tab labeled "Console 1/A". The console displays three directory listings in Chinese:

```
檔案路徑： D:/PythonBook/附書光碟/ch04/oswalk  
目錄串列： ['Dir']  
檔案串列： ['oswalk.py', 'oswalk1.txt']  
  
檔案路徑： D:/PythonBook/附書光碟/ch04/oswalk\Dir  
目錄串列： ['SubDir']  
檔案串列： ['Dir1.txt', 'Dir2.txt']  
  
檔案路徑： D:/PythonBook/附書光碟/ch04/oswalk\Dir\SubDir  
目錄串列： []  
檔案串列： ['SubDir1.txt']
```


4.1.4 shutil

常用的方法如下：

/The commonly used methods are as follows:

//Metode yang umum digunakan adalah sebagai berikut:

屬性或方法	說明
copy(src,dst)	複製 src 檔案為 dst 檔
copytree(src,dst)	將 src 目錄及目錄中所有檔案複製到 dst
rmtree(dir)	刪除 dir 目錄和目錄中所有檔案
move(src,dst)	將 src 目錄或檔案搬移到 dst

例如：複製 <shutil.py> 為 <newfile.py> 檔。(<shutil.py>)

/For example: Copy <shutil.py> to the <newfile.py> file.

//Sebagai contoh: Salin <shutil.py> ke file <newfile.py>.

```
import os,shutil
cur_path=os.path.dirname(__file__) # 取得目前路徑
destfile= cur_path + "\\\" + "newfile.py"
shutil.copy("shutil.py",destfile ) # 檔案複製
```

4.1.5 glob

語法： /grammar: //tatabahasa:

```
glob.glob("路徑名稱")
```

路徑名稱可以明確指定檔案名稱，也可使用「*」萬用字元。

/The path name can be specified with the file name explicitly, or the "*" universal character can be used.

//Nama jalan dapat ditentukan dengan nama file secara eksplisit, atau karakter universal "*" dapat digunakan.

例如：取得<glob.py> 檔、檔名前兩個字元是 os 開頭的所有 py 檔案以及所有副檔名為 txt 的檔案。(<glob.py>)

/For example: Get the <glob.py> file, the first two characters of the file name are all py files starting with os and all files with the file name txt.

//Sebagai contoh: Dapatkan file <glob.py>, dua karakter pertama dari nama file adalah file py yang dimulai dengan os dan semua file dengan nama file txt.

```
import glob
files = glob.glob("glob.py") + glob.glob("os*.py") + glob.glob("*.txt")
for file in files:
    print(file)
```