

# Chapter 8

---

TIME

# 時間

---

Python 是以 tick 做為時間的計數單位，準確度可以到百萬分之一秒

tick 是以微秒為單位的 float 數值，1 秒 = 1,000,000 ticks

Python 時間算法是從 1970 年 1 月 1 日 0 點起算到現在總共經過多少秒

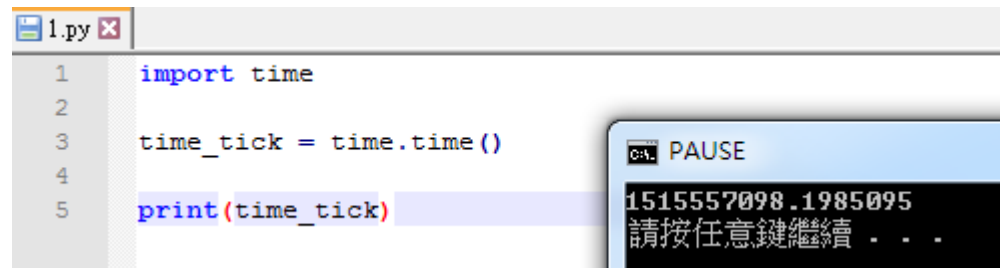
Note : ticks 總數是照格林威治標準時間，不是照使用者的時區

# 時間

---

使用 time function 需要 import 時間模組（下一章會進一步教學）

time\_tick 取到的時間為從 1970, 1-1, 0:00:00 開始到現在經過的秒數

A screenshot showing a Python script in a text editor and its execution output. The script, named '1.py', contains five lines of code: 'import time', 'time\_tick = time.time()', and 'print(time\_tick)' on lines 1, 2, and 5 respectively. The output window shows the result '1515557098.1985095' and a prompt '請按任意鍵繼續...' (Press any key to continue...).

只有得到幾秒是沒辦法馬上換算到現在的時間

接下來會要我們會學會如何使用 time 模組中的各種函數

可以順利的將得到的時間秒數轉換成為常見的時間模式

# 時間

---

## 時間模組

`gmtime([second])`

`second` 是選擇性的參數，可以是 `int` 或 `float` 型態

使用 `gmtime()` 這個函數得到從 1970 年 1 月 1 日 0 點開始到所傳入秒數的時間

如果沒有參數，預設是目前的時間

Note: 目前的時間是格林威治標準時間，而非當地時間 (如台灣)

# 時間

gmtime() 回傳的格式是 **tuple** ( time.struct\_time( .... ) )

```
1 import time
```

```
2  
3 icy = time.gmtime()
```

```
4 icy1 = time.gmtime(123.45)
```

```
5 icy2 = time.gmtime(12345)
```

```
6  
7 print(icy)
```

```
8 print("")
```

```
9 print(icy1)
```

```
10 print("")
```

```
11 print(icy2)
```

Ctrl PAUSE

```
time.struct_time(tm_year=2018, tm_mon=1, tm_mday=14, tm_hour=2, tm_min=51, tm_sec=50, tm_wday=6, tm_yday=14, tm_isdst=0)
```

```
time.struct_time(tm_year=1970, tm_mon=1, tm_mday=1, tm_hour=0, tm_min=2, tm_sec=3, tm_wday=3, tm_yday=1, tm_isdst=0)
```

```
time.struct_time(tm_year=1970, tm_mon=1, tm_mday=1, tm_hour=3, tm_min=25, tm_sec=45, tm_wday=3, tm_yday=1, tm_isdst=0)
```

請按任意鍵繼續 . . .

# 時間

```
C:\ PAUSE
time.struct_time(tm_year=2018, tm_mon=1, tm_mday=14, tm_hour=2, tm_min=51, tm_sec=50, tm_wday=6, tm_yday=14, tm_isdst=0)
```

Attribute	Value
tm_year	year
tm_mon	1 to 12
tm_mday	1 to 31
tm_hour	0 to 23
tm_min	0 to 59
tm_sec	1 to 61
tm_wday	0 (Mon) to 6
tm_yday	1 to 366
tm_isdst	-1, 0, or 1

# 時間

---

## 時間模組

`localtime([second])`

(`localtime` 和 `gmtime` 很像)

`second` 是選擇性的參數，可以是 `int` 或 `float` 型態

使用 `localtime()` 這個函數得到從 1970 年 1 月 1 日 0 點開始到所傳入秒數的時間

如果沒有參數，預設是目前”時區”的時間

# 時間

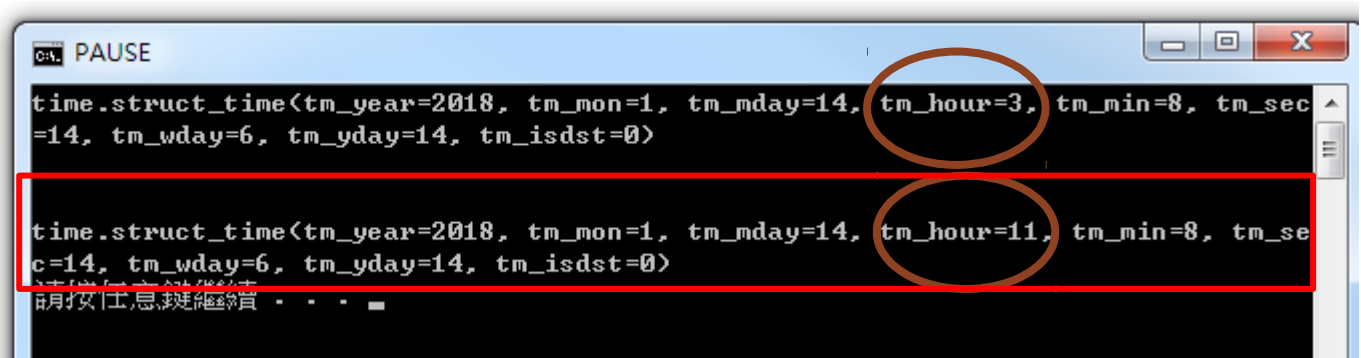
localtime() 回傳的時間是目前所在地時區的時間（台灣：GMT +8）

```
import time

icy = time.gmtime()

icy1 = time.localtime()

print(icy)
print("")
print("")
print(icy1)
```



```
PAUSE
time.struct_time(tm_year=2018, tm_mon=1, tm_mday=14, tm_hour=3, tm_min=8, tm_sec=14, tm_wday=6, tm_yday=14, tm_isdst=0)
time.struct_time(tm_year=2018, tm_mon=1, tm_mday=14, tm_hour=11, tm_min=8, tm_sec=14, tm_wday=6, tm_yday=14, tm_isdst=0)
請按任意鍵繼續 . . .
```



# 時間

---

## 時間模組

`asctime([tuple_variable])`

`asctime` 的功能是將輸入的時間轉換成時間日期的字串

`tuple_variable` 是選擇性的參數，必需是 **tuple** 型態

參數的順序為年、月、日、時、分、秒、星期幾、第幾天、是否日光節約時間

如果沒有參數，預設是目前”時區”的時間 (`time.localtime()`)

# 時間

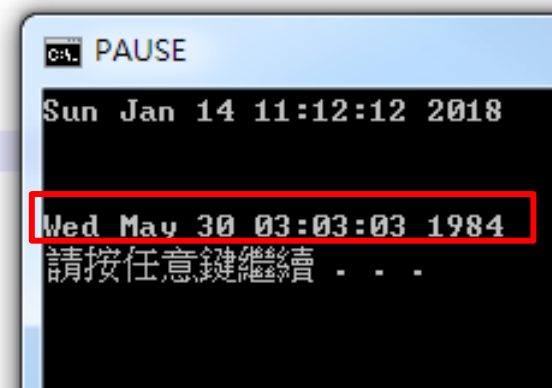
asctime 的功能是將輸入的時間轉換成時間日期的字串回傳

```
import time

icy = time.asctime(time.localtime())

icy1 = time.asctime( (1984, 5, 30, 3, 3, 3, 2, 3, 0) )

print(icy)
print("")
print("")
print(icy1)
```



# 時間

## 時間模組

`ctime([seconds])`

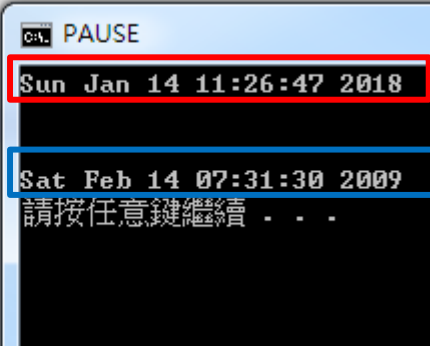
`ctime` 的功能是將輸入的 `int` 或 `float` 秒數轉換成時間日期的字串再回傳

`seconds` 是選擇性的參數，可以是 `int` 或 `float` 型態

如果沒有參數，預設會以是“目前”時區”的時間 (`time.localtime()`) 轉成秒數

```
import time
icy = time.ctime()
icy1 = time.ctime(1234567890)

print(icy)
print("")
print("")
print(icy1)
```



# 時間

---

## 時間模組

### strftime()

strf 接收 tuple 型態的參數，並依所需的格式設定時間的格式

strftime 的功能是時間函數的格式化輸出

```
import time  
  
icy = time.strftime("%m-%d, %Y", time.localtime() )  
  
print(icy)  
print("")
```

PAUSE

01-14, 2018

請按任意鍵繼續 . . .

# 時間

幾種常用的 `strftime` 的**格式化符號** (`%+Symbol`)

Symbol	Meaning
%y	年份 ( 00-99 )
%Y	年份 ( 000-9999 )
%m	月份 ( 01-12 )
%d	天 ( 0-31 )
%H	小時 ( 0-23 )
%I	小時 ( 01-12 )
%M	分鐘 ( 00-59 )
%S	秒 ( 00-59 )

Symbol	meaning
%a	星期幾的縮寫 (Mon, Tue, ...)
%A	星期幾的完整名稱 (Monday)
%b	月份名的縮寫 (Jan, Dec)
%B	月份名 (January, December)
%c	完整的日期和時間
%j	是一年內的某一天 ( 001-366 )
%p	顯示 AM or PM.
%U	第幾週 ( 00-53 ) 星期天起始點
%w	星期幾 ( 0-6 ) ，星期天為星期的開始
%W	第幾週 ( 00-53 ) 星期一起始點
%x	完整日期

# 時間

---

來一個一個試 strftime 的格式化功能吧

(%y, %Y, %m, %d )

```
import time

icy = time.strftime("%y, %Y, %m, %d", time.localtime() )

print(icy)
print("")
```



18, 2018, 01, 14

請按任意鍵繼續 . . .

# 時間

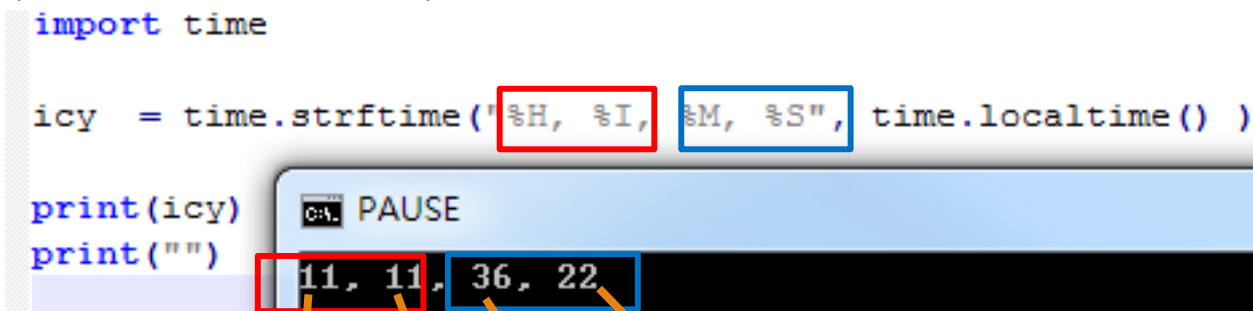
來一個一個試 strftime 的格式化功能吧

(%H, %I, %M, %S )

```
import time

icy = time.strftime('%H, %I, %M, %S', time.localtime() )

print(icy)
print("")
```



11, 11, 36, 22

11 點, 11 點, 36 分, 22 秒

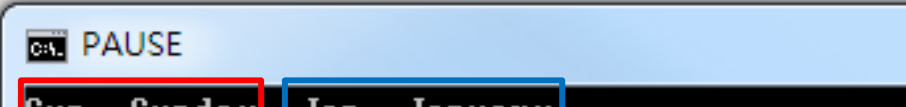
# 時間

---

來一個一個試 strftime 的格式化功能吧

(%a, %A, %b, %B )

```
import time  
  
icy = time.strftime("%a, %A, %b, %B", time.localtime() )  
  
print(icy)  
print("")
```





# 時間

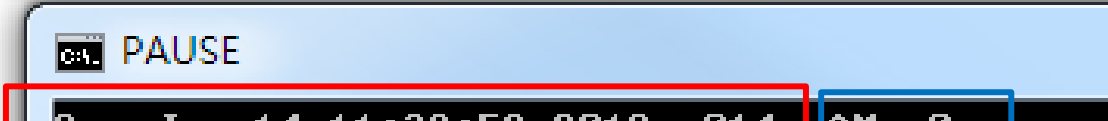
來一個一個試 strftime 的格式化功能吧

(%c, %j, %p, %w )

```
import time

icy = time.strftime("%c, %j, %p, %w", time.localtime() )

print(icy)
print("")
```



完整時間，今天是一年的第幾天， PM/AM, 星期幾

# 時間

## 時間模組

`strptime()` [ **string format** ]

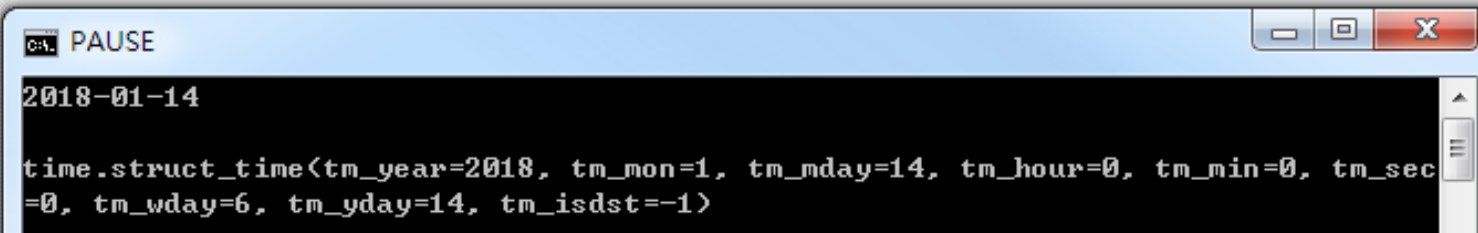
`strf` 接收接收一個字串參數與一個格式參數，然後會回傳時間的 tuple

`strptime` 的功能剛好跟 `strftime` 相反

```
import time

icy = time.strftime("%Y-%m-%d", time.localtime() )
print(icy)
print("")

icy1 = time.strptime(icy, "%Y-%m-%d")
print(icy1)
print("")
```

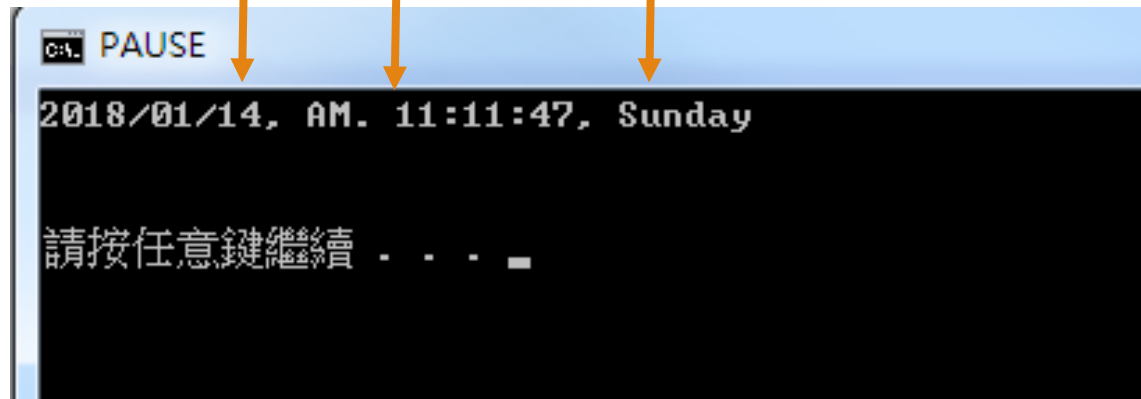


The screenshot shows a Python interpreter window titled "PAUSE". The output of the code is displayed in a black terminal window. The first line of output is "2018-01-14". The second line of output is a tuple representing the parsed time: "time.struct\_time(tm\_year=2018, tm\_mon=1, tm\_mday=14, tm\_hour=0, tm\_min=0, tm\_sec=0, tm\_wday=6, tm\_yday=14, tm\_isdst=-1)".

# 隨堂練習

---

1. 輸出現在的日期、時間、與星期幾
2. 格式如下所示



```
C:\> PAUSE
2018/01/14, AM. 11:11:47, Sunday
請按任意鍵繼續 . . . _
```

Any Questions !?