Python程式設計

陳裔專
ycchenroc@gmail.com
2017/11/29

Python程式語言誕生於1989年

- ✓創始人Guido van Rossum(吉多．范羅蘇姆)提出
- ✓迄今有二十多年的歷史
- ✓它是高階語言
  - ➢支援物件導向
  - ➢語言本身能跨越平台，無論是Linux、Mac或者是Windows皆能暢行無阻

Python技術及開發環境介紹

Python語言的特色

✓Python型別是動態、強型別

- 程式碼執行才會檢查
- 不同的資料型別採用高標準規範；比如數字加字串是不會執行的

✓Python是腳本亦是程式語言

- 腳本語言是一種解釋性的語言，不需要編譯，可以直接用，由解釋器來負責解釋。
- 除了本身擁有功能完備的標準函式庫，也能加入第三方函式庫輕鬆完成任務

Python版本發展

版本	簡介
2.0	2000年10月16日發布，支援Unicode和垃圾回收機制
2.7	2010年7月3日發行
3.0	2008年12月3日發布，此版不完全相容之前的Python原始碼
3.5	發布2015年9月13日

Python 2.x與3.x版同時存在

- ✓Python語言有趣之處卻是Python 2x和Python 3x同時存在，而彼此之間並非完全相容
  - ➢Python 2.7是Python官方於2x系列所發表的最後版本，由於資源較豐富，第三方函式庫以它為基底依然不少
- ✓為什麼用Python 3
  - ➢更強大更方便的語法
  - ➢更強大更方便的函式庫
  - ➢Python 3 是目前官方持續開發（加新功能）的版本。

Python應用領域有哪些

- ✓遊戲、網頁開發
- ✓科學計算
- ✓資料庫支援
- ✓企業級應用-ERP、CRM
- ✓機器人、AI、物聯網等

- ✓Python 是用於大數據分析非常火紅的程式語言，從資料的取、分析與視覺化顯示皆是一把罩。
- ✓若想要了解目前最受歡迎的程式
- ✓語言，可上網至https://www.tiobe.com/tiobe-index/，即可知目前較受歡迎的程式語言排行榜。
- ✓ 下圖是2017 年6 月程式語言受歡迎程度的排行榜。

2017年6月程式語言歡迎程度排行榜

Jun 2017	Jun 2016	Change	Programming Language	Ratings
1	1		Java	14.493%
2	2		C	6.848%
3	3		C++	5.723%
4	4		Python	4.333%
5	5		C#	3.530%
6	9	^	Visual Basic .NET	3.111%
7	7		JavaScript	3.025%
8	6	v	PHP	2.774%
9	8	v	Perl	2.309%
10	12	^	Assembly language	2.252%
11	10	v	Ruby	2.222%
12	14	^	Swift	2.209%
13	13		Delphi/Object Pascal	2.158%
14	16	^	R	2.150%
15	48	^^	Go	2.044%

Python自學資源

- ✓ Python 教學

  https://docs.python.org.tw/3/tutorial/

- ✓ 語言技術：Python Gossip

  http://openhome.cc/Gossip/Python/index.html

- ✓ 用Python玩轉數據 Data Processing Using Python

  https://www.coursera.org/learn/hipython

- ✓ Python 台灣使用者群組

  http://wiki.python.org.tw/

- ✓ Python 3.6.3 documentation

  https://docs.python.org/3/

- ✓ Python 2.7.14 documentation

  https://docs.python.org/2/

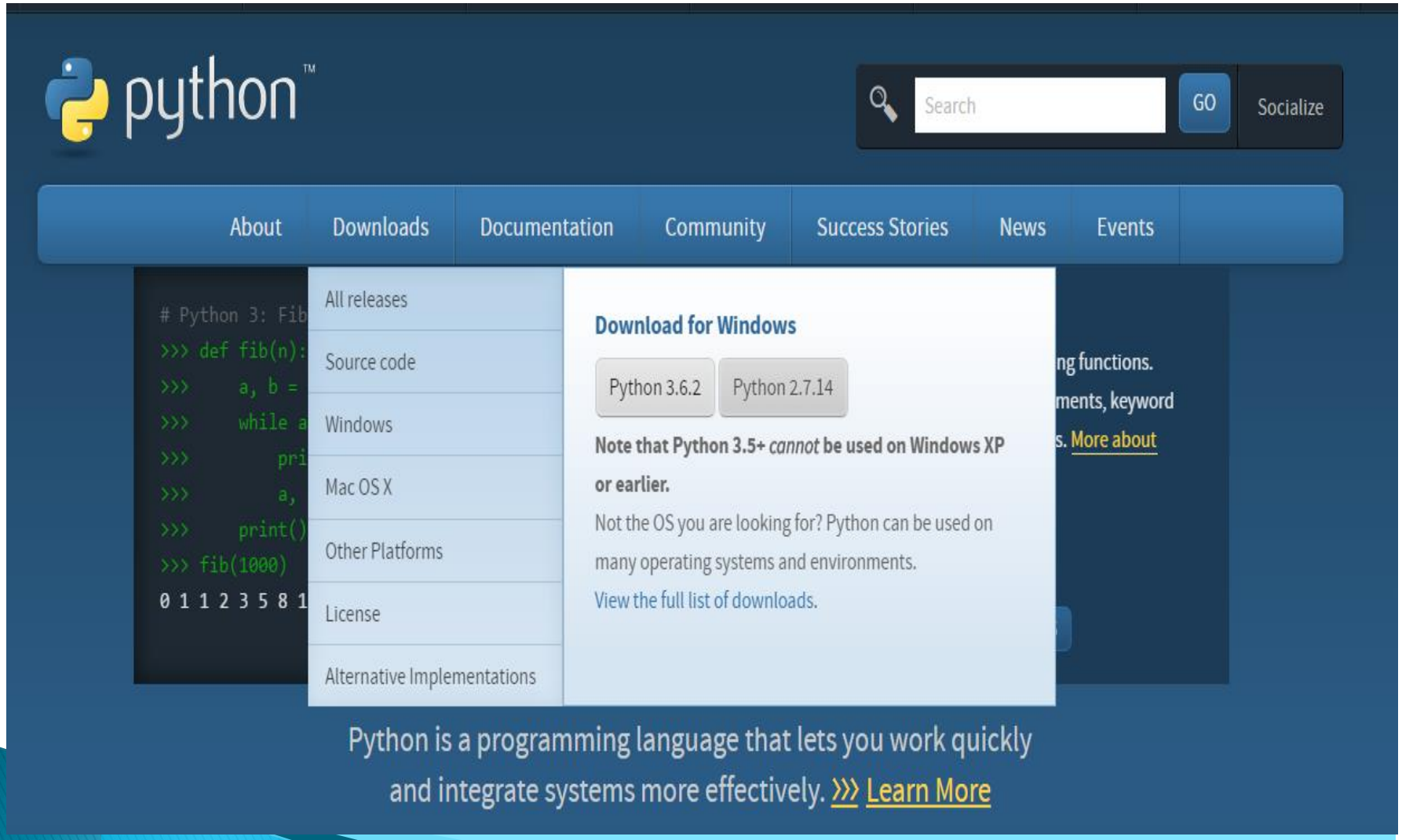
- ✓ Python 線上中文資源

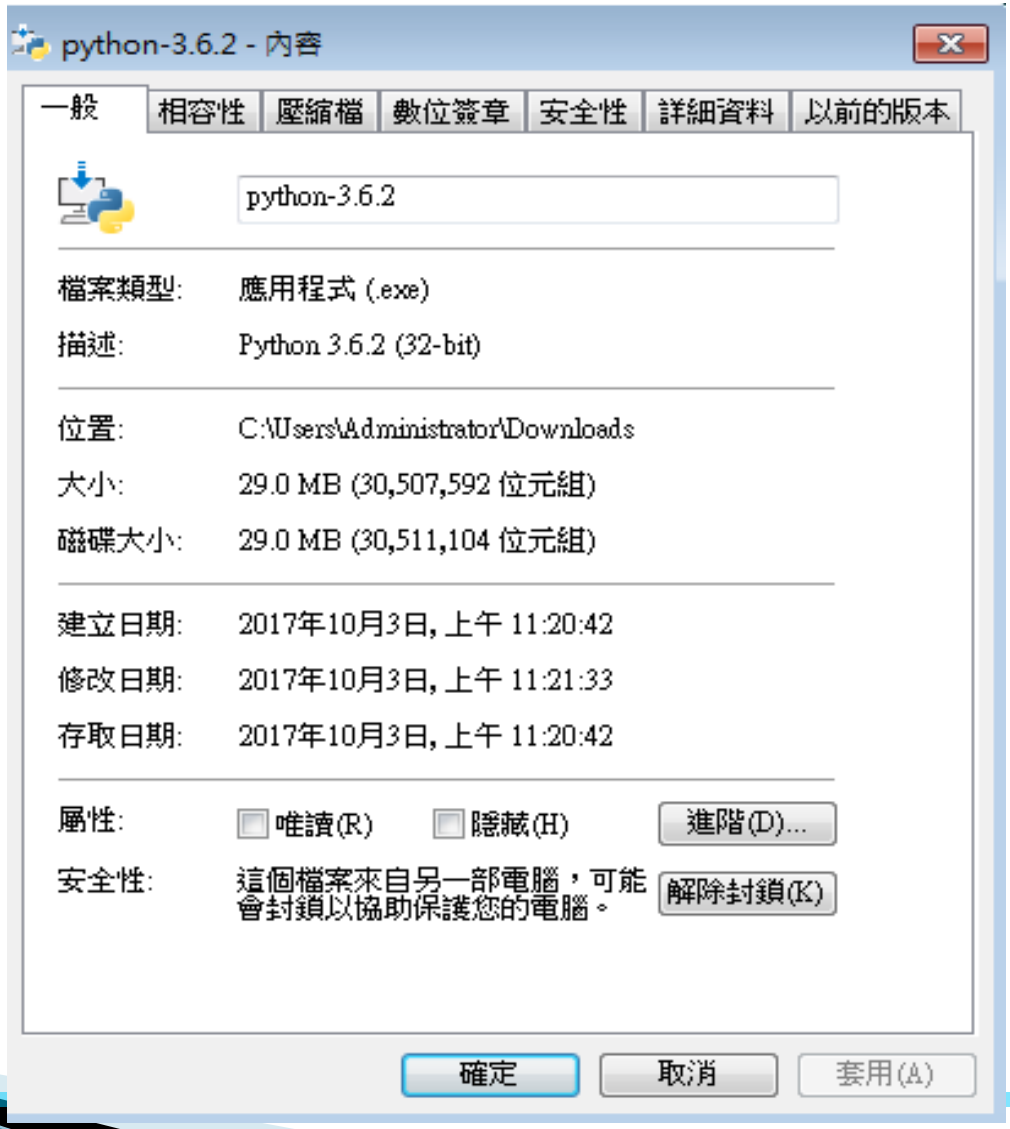
  http://www.kaiching.org/2012/10/python-resource.html

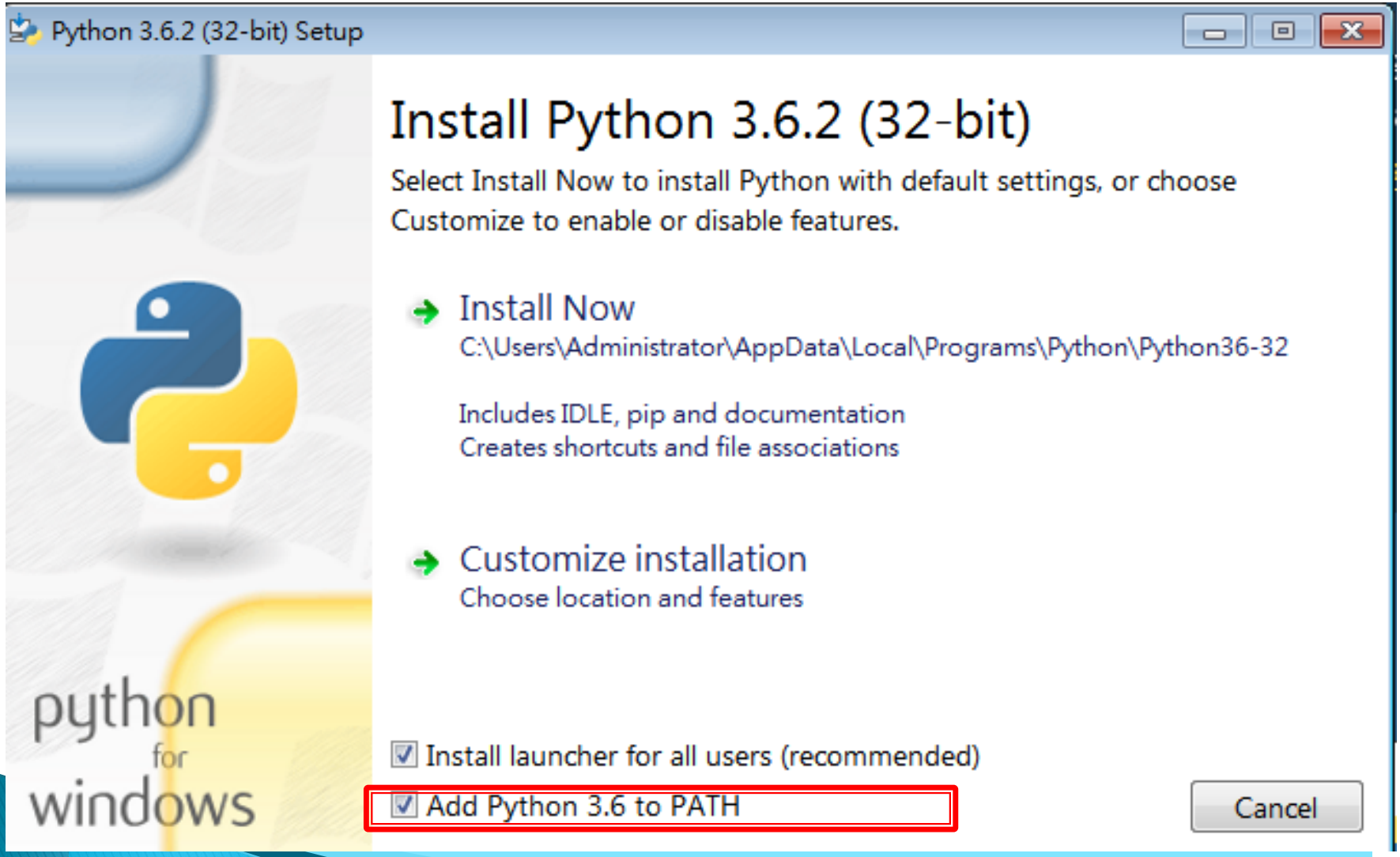
Python 安裝

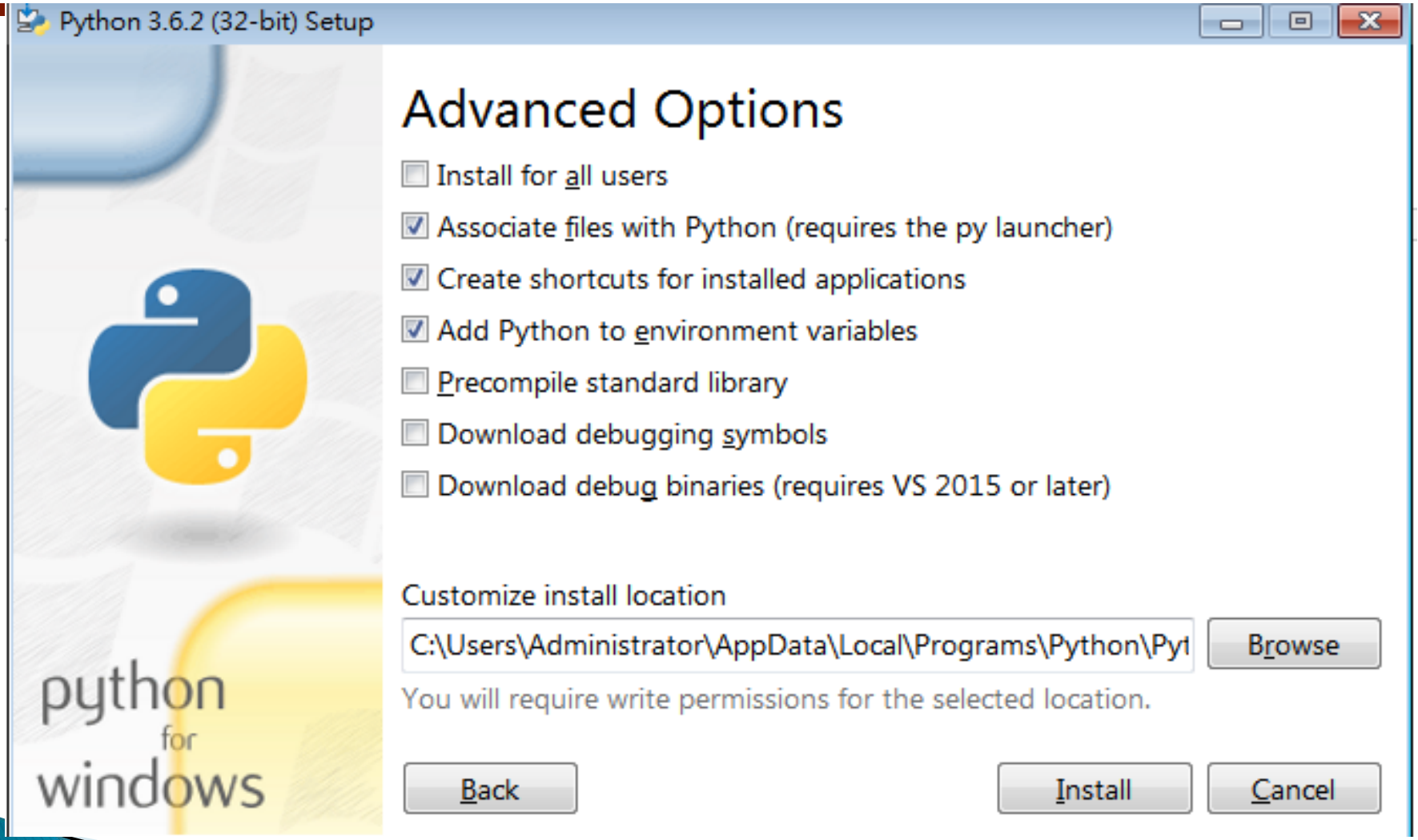
- ✓官方網站
  - https://www.python.org/

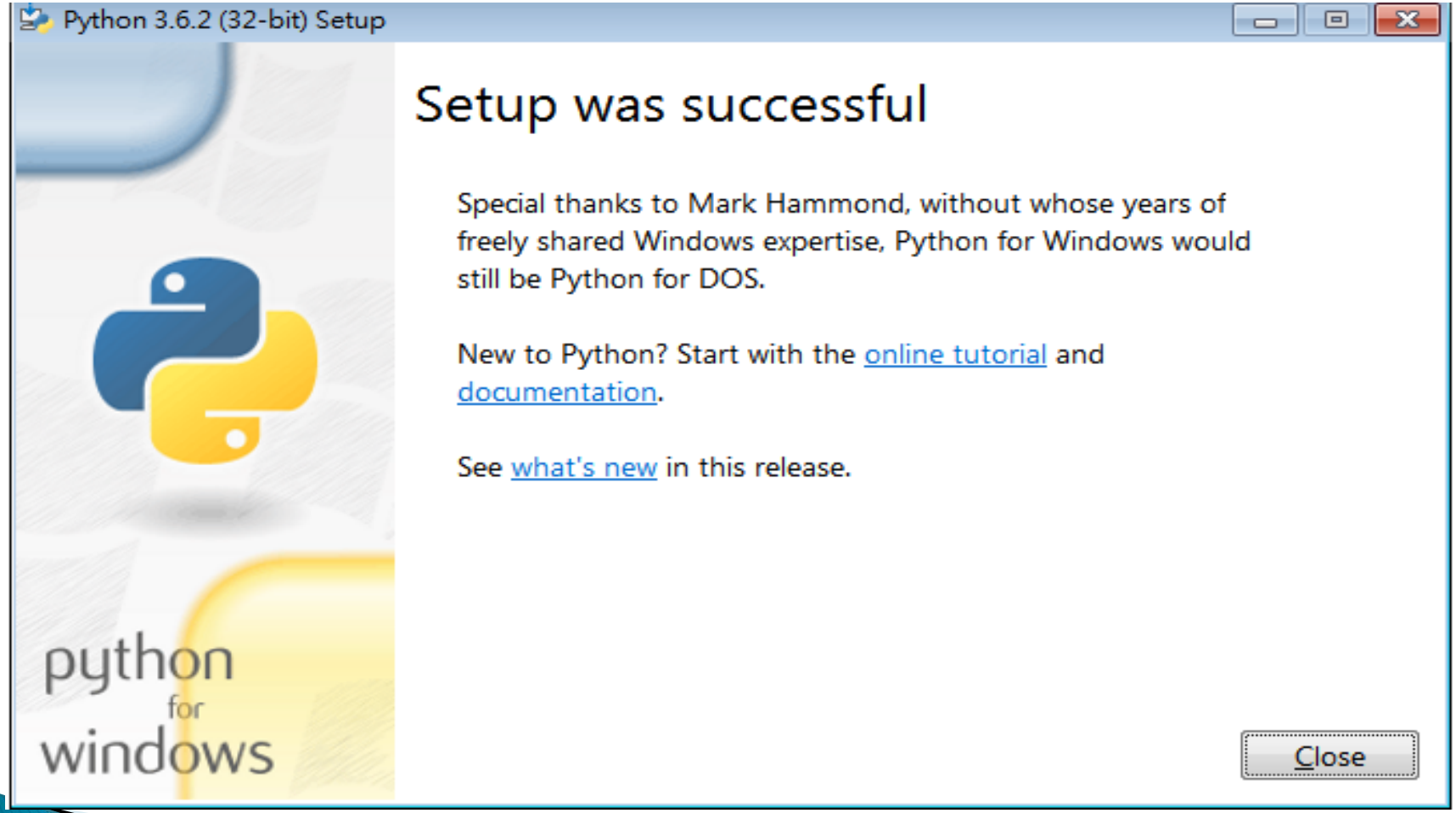
下載與安裝 Python 3.5




python-3.6.2 - 內容
一般
相容性
壓縮檔
數位簽章
安全性
詳細資料
以前的版本
python-3.6.2
檔案類型:
應用程式 (.exe)
描述:
Python 3.6.2 (32-bit)
位置:
C:\Users\Administrator\Downloads
大小:
29.0 MB (30,507,592 位元組)
磁碟大小:
29.0 MB (30,511,104 位元組)
建立日期:
2017年10月3日, 上午 11:20:42
修改日期:
2017年10月3日, 上午 11:21:33
存取日期:
2017年10月3日, 上午 11:20:42
屬性:
唯讀(R)
隱藏(H)
進階(D)...
安全性:
這個檔案來自另一部電腦，可能會封鎖以協助保護您的電腦。
解除封鎖(K)
確定
取消
套用(A)


Python 3.6.2 (32-bit) Setup
Install Python 3.6.2 (32-bit)
Select Install Now to install Python with default settings, or choose
Customize to enable or disable features.
Install Now
C:\Users\Administrator\AppData\Local\Programs\Python\Python36-32
Includes IDLE, pip and documentation
Creates shortcuts and file associations
Customize installation
Choose location and features
python
for
windows
Install launcher for all users (recommended)
Add Python 3.6 to PATH
Cancel


Python 3.6.2 (32-bit) Setup
Advanced Options
Install for all users
Associate files with Python (requires the py launcher)
Create shortcuts for installed applications
Add Python to environment variables
Precompile standard library
Download debugging symbols
Download debug binaries (requires VS 2015 or later)
Customize install location
C:\Users\Administrator\AppData\Local\Programs\Python\Pyt
Browse
You will require write permissions for the selected location.
python
for
windows
Back
Install
Cancel

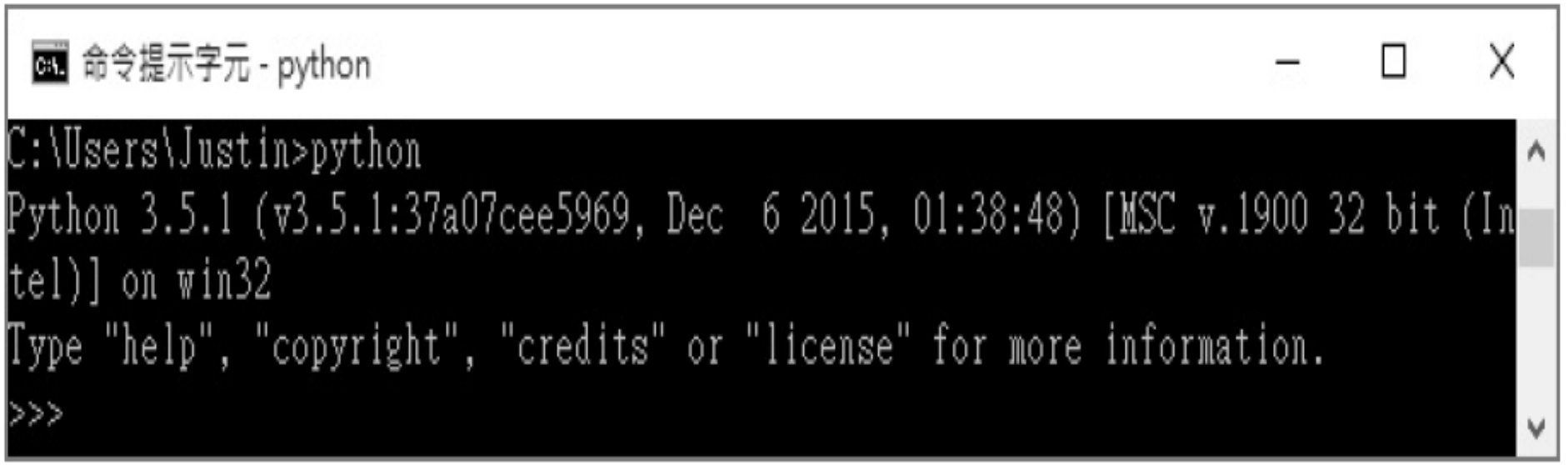

Python 3.6.2 (32-bit) Setup
Setup was successful
Special thanks to Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.
New to Python? Start with the online tutorial and documentation.
See what's new in this release.
python for windows
Close

2 從 REPL 到 IDE

- 學習目標
  - ✓ 使用 REPL
  - ✓ 設定原始碼檔案編碼
  - ✓ 認識 IDE 的使用

使用 REPL

✓REPL（Read-Eval-Print Loop，又稱為Python Shell）是一個簡單的，交互式的編程環境。



使用 REPL

```
>>> 1 + 2
3
>>> _
3
>>> 1 + _
4
>>> _
4
>>>
```

```
>>> 'Hello World'
'Hello World'
>>> print(_)
Hello World
>>> print('Hello World')
Hello World
>>>
```

```
>>> print 'Hello World'
  File "<stdin>", line 1
    print 'Hello World'
                      ^
SyntaxError: Missing parentheses in call to 'print'
>>>
```

取得協助訊息

```
>>> help()

Welcome to Python 3.5's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at http://docs.python.org/3.5/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help>
```

```
help> keywords

Here is a list of the Python keywords.  Enter any keyword to get more help.

False               def                 if                  raise
None                del                 import              return
True                elif                in                  try
and                 else                is                  while
as                  except              lambda              with
assert              finally             nonlocal            yield
break               for                 not
class               from                or
continue            global              pass

help>
```

```
help> print
Help on built-in function print in module builtins:

print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
    Optional keyword arguments:
    file:  a file-like object (stream); defaults to the current sys.stdout.
    sep:   string inserted between values, default a space.
    end:   string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.

help>
```

```
help> quit

You are now leaving help and returning to the Python interpreter.
If you want to ask for help on a particular object directly from the
interpreter, you can type "help(object)".  Executing "help('string')"
has the same effect as typing a particular string at the help> prompt.
>>> help(print)
Help on built-in function print in module builtins:

print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    ...略

>>>
```

✓如果要離開 REPL 環境，可以執行 quit() 函式

```
C:\Users\nou>python
Python 3.6.2 (v3.6.2:5fd33b5, Jul  8 2017, 04:14:34) [MSC v.1900 32 bit (Intel)]
 on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> quit()

C:\Users\nou>
```

- ✓>>> 是Python敘述的提示，此處可以鍵入敘述。
- ✓現在鍵入print('Learning Python now!')，然後按下Enter鍵。
- ✓字串Learning Python now! 將會出現在螢幕上，如下所示：

```
>>> print('Learning Python now!')
Learning Python now!
```

- ✓字串（string）是程式設計述語，表示一系列的字元。
- ✓Python可以雙引號或單引號括起字串，由於單引號不需要按Shift鍵，所以可較快速鍵入完成。
- ✓Python不會在輸出結果顯示這些引號。

輸出函式print()

- ✓print敘述是Python的內建函式，用來將字串顯示於螢幕。
- ✓函式是用來執行動作的。此處的print函式是顯示一訊息於螢幕上。
- ✓在程式設計的述語中，當您使用一函式，可以"呼叫一函式"。

Hello.py
Welcome.py

✓接下來鍵入print('Python is fun')，然後按下Enter鍵。字串Python is fun將會出現在螢幕，如下所示。

```
>>> print('Python is fun')
Python is fun
```

- ✓如果要顯示：

  Let’s go

- ✓程式碼不能撰寫成

  print('Let's go')

- ✓因為第一個單引號會與Let 下的單引號結合在一起，導致最後在go 的單引號沒有匹配的對象，故會產生錯誤的訊息。
- ✓要解決此一問題必須藉助轉義序列（escape sequence）。
- ✓轉義序列是由反斜線加上特定的字元所組成的，其用來執行某一特定的動作。如表2-1 所示：

表2-1　轉義序列

轉義序列	功能說明
\n	換行
\t	跳八格
\\	輸出反斜線
\"	輸出雙引號
\'	輸出單引號

✓從表2-1得知，可使用 \' 符號，用以印出單引號。正確的寫法如下：

```
print('Python is fun')
print('Let\'s go')
```

- ✓不要將任何標點符號，如分號，放在敘述的尾端。例如以下的程式碼，Python也會產生錯誤的訊息。

```
#display two messages
print('Learning Python now!');
print('Python is fun');
```

- ✓Python程式大、小寫字母是有差別的。若將程式中的print改為Print將會產生錯誤。

建立Python的原始碼檔案

✓在 >>> 提示下輸入敘述是很方便的，但無法儲存。為了能夠儲存以便日後使用，您可以在IDLE選單（如圖所示）

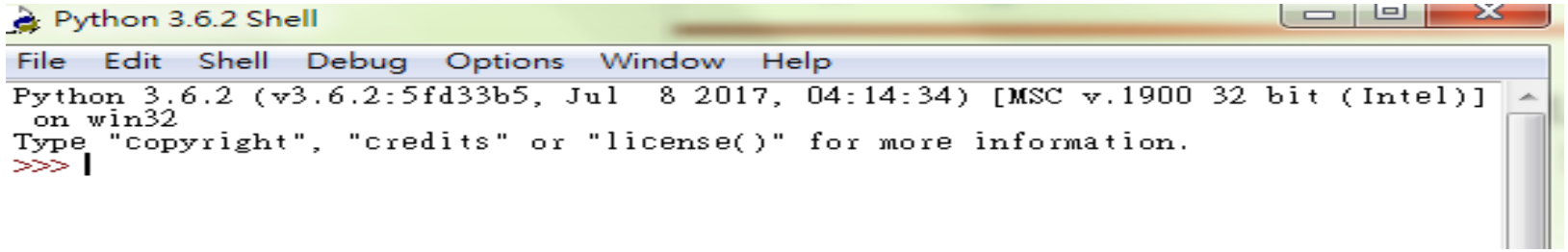
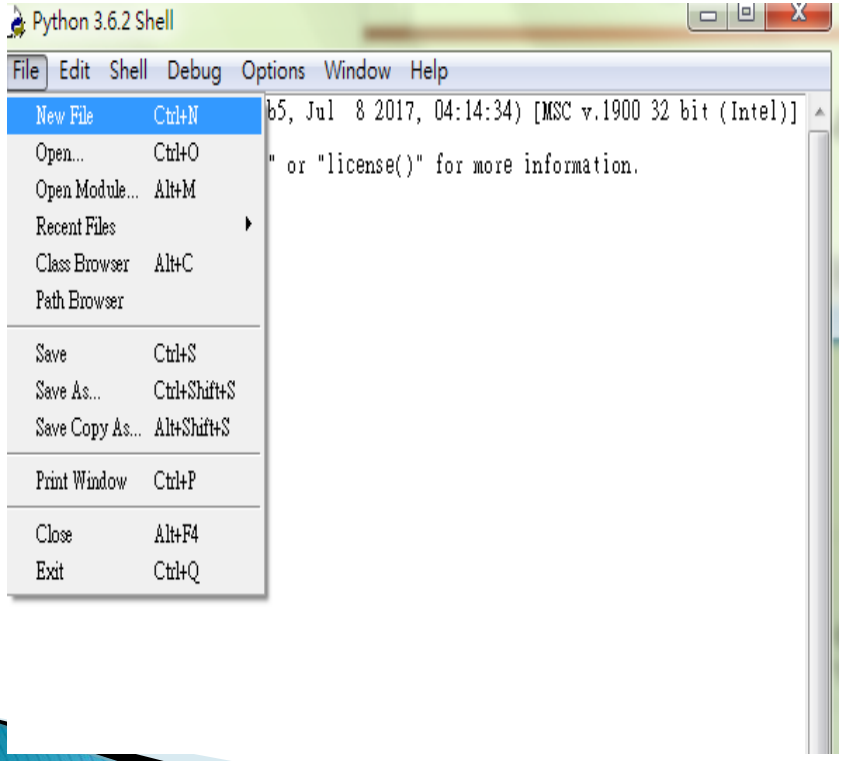
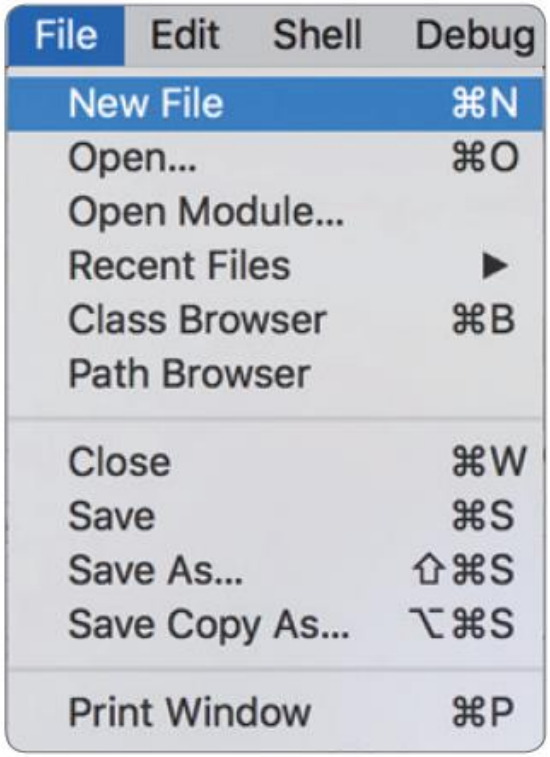


圖1-7 IDLE選單

✓選取File->New File（如圖1-8）來建立原始碼（source code）檔案。



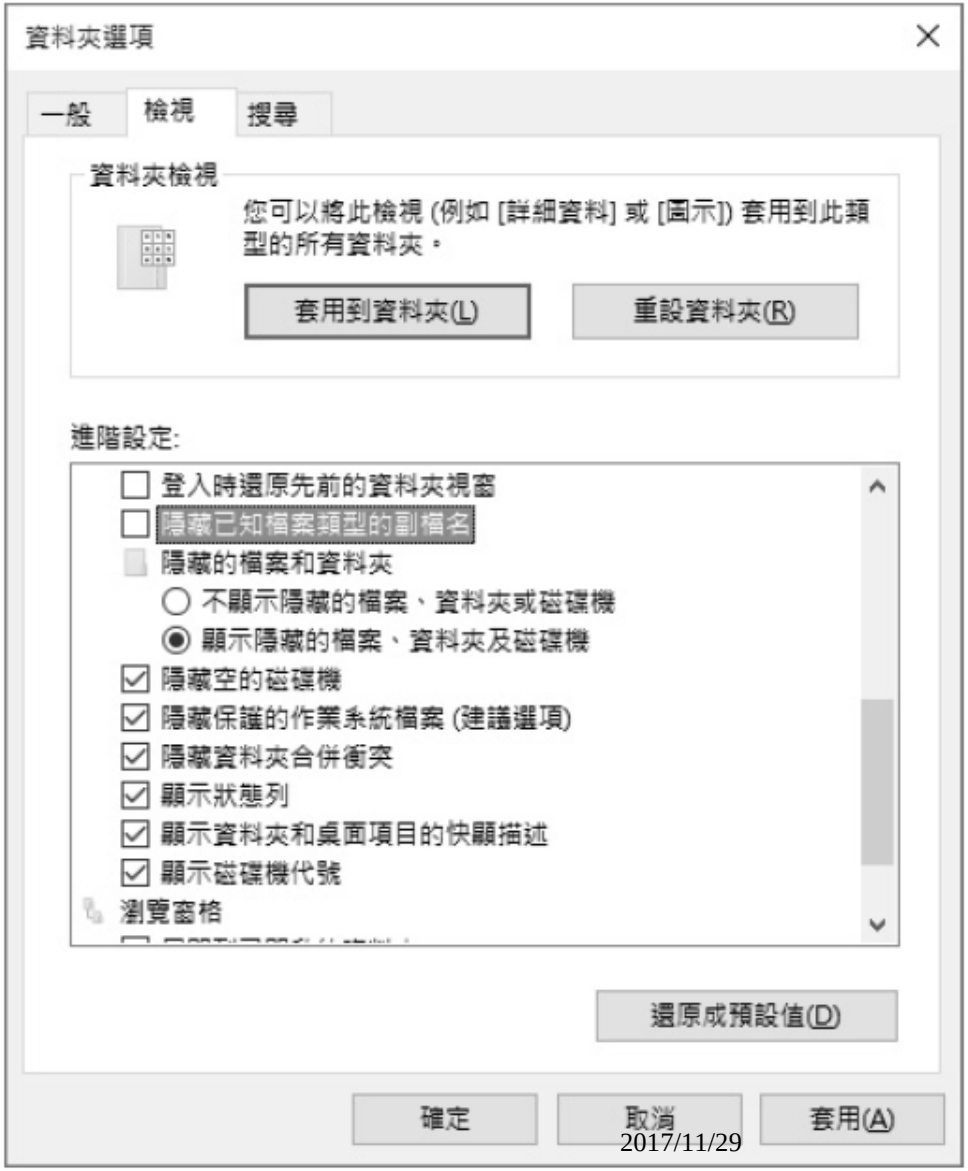


● 圖1-8　建立一新檔的選單

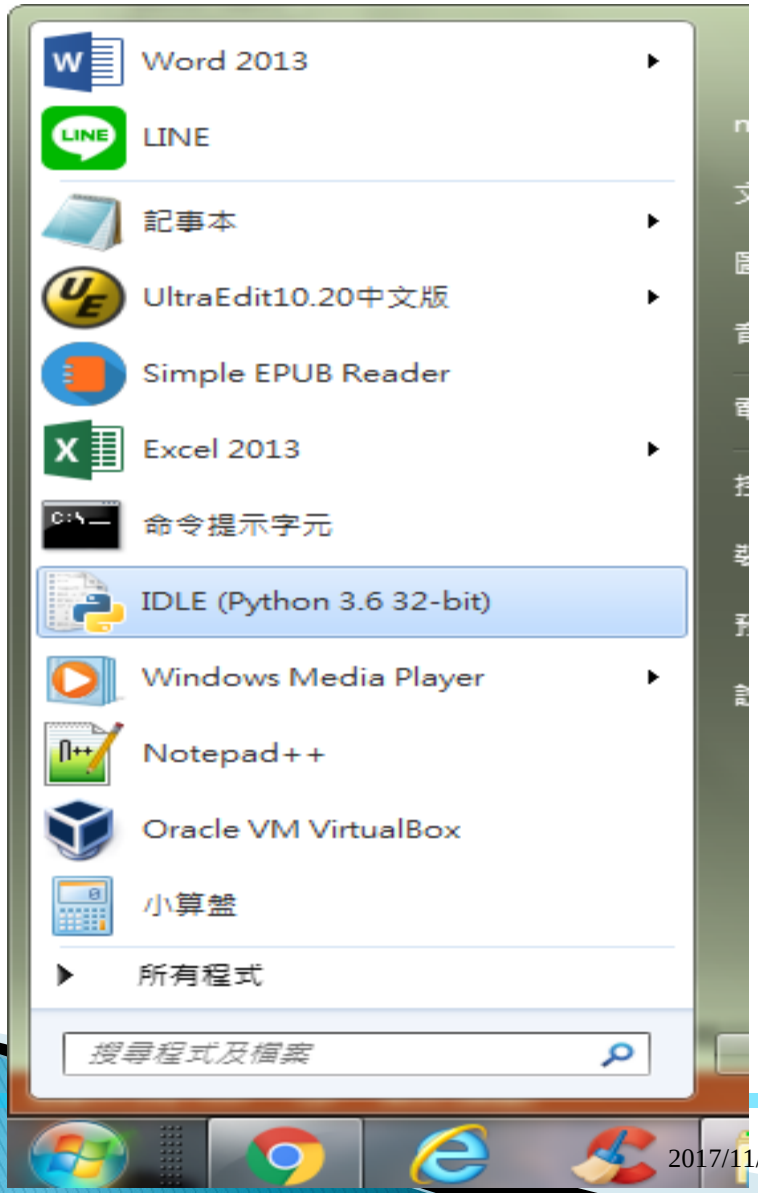
撰寫 Python 原始碼

- 在正式撰寫程式之前, 請先確定你可以看到檔案的副檔名
- Python的延伸檔名是 .py
- 如果目前在「檔案總管」下無法看到副檔名,Windows 7 下請執行「組合管理/資料夾和搜尋選項」，Windows 8 或10 都可以執行「檢視/選項」，之後都是切換至「檢視」頁籤，將「隱藏已知檔案類型的副檔名」選項取消。

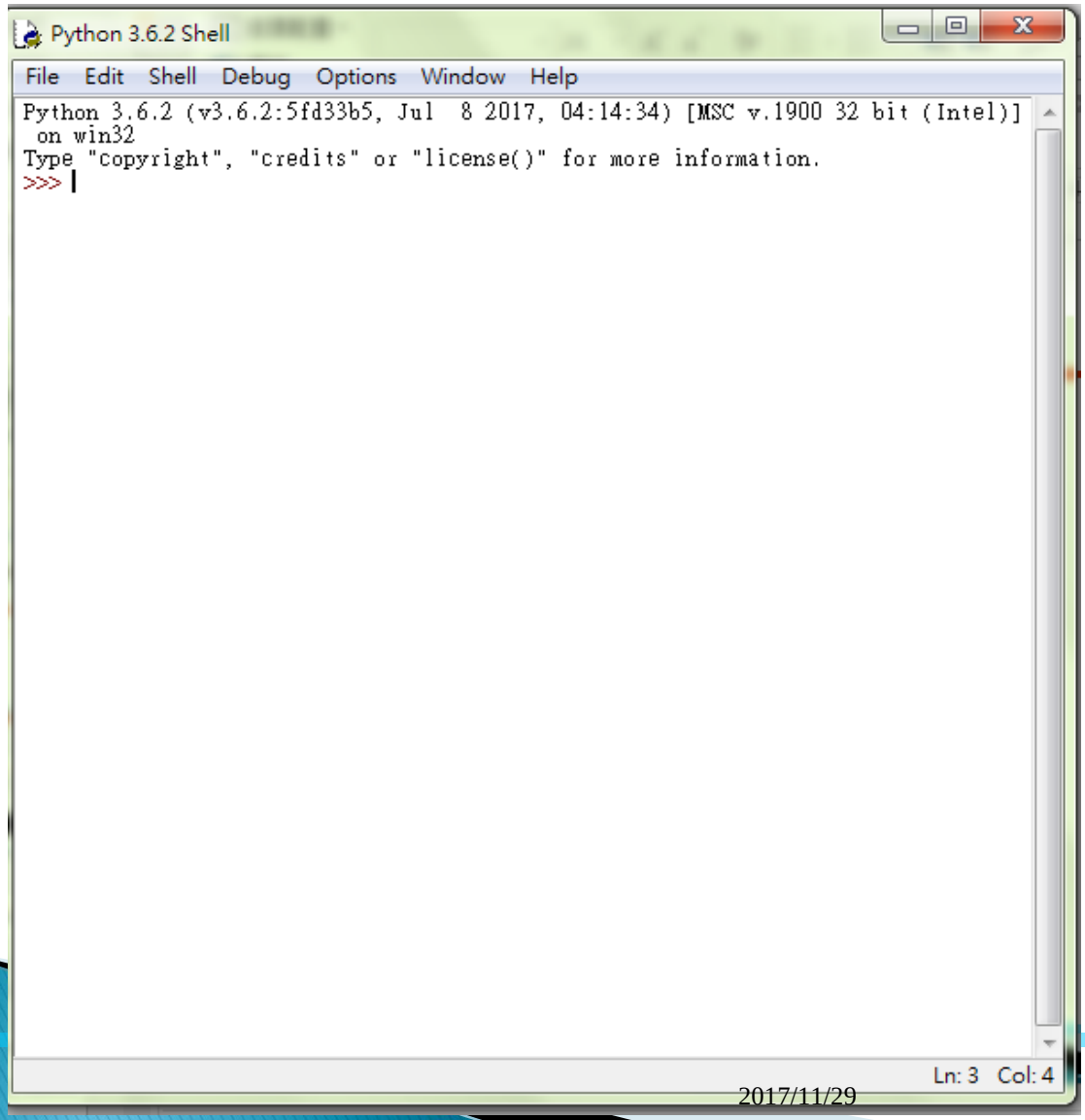
撰寫 Python 原始碼



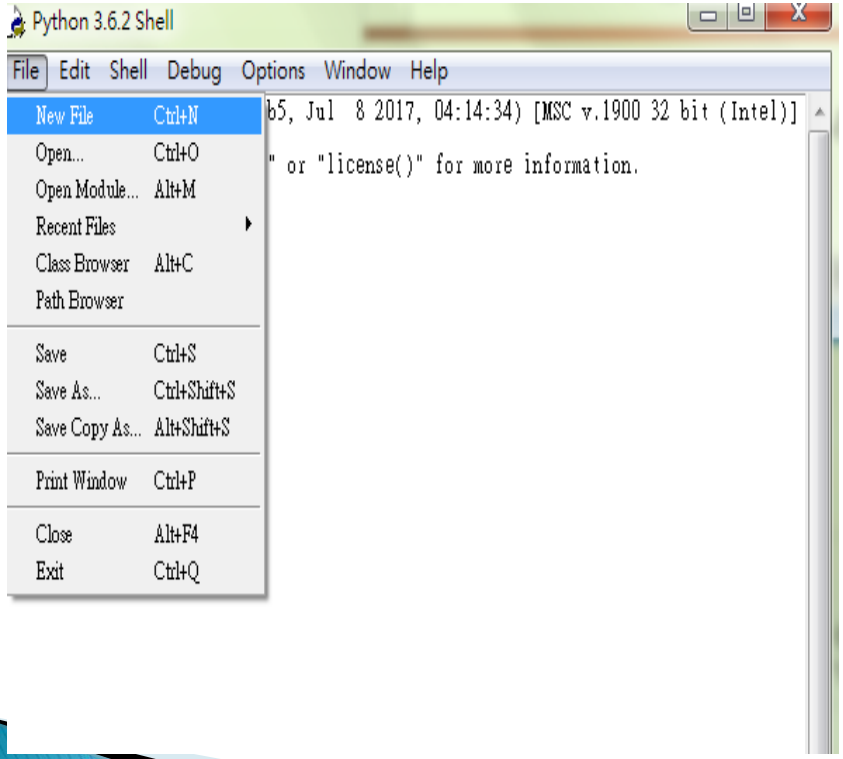
使用 IDLE


Word 2013
LINE
記事本
UltraEdit10.20中文版
Simple EPUB Reader
Excel 2013
命令提示字元
IDLE (Python 3.6 32-bit)
Windows Media Player
Notepad++
Oracle VM VirtualBox
小算盤
所有程式
搜尋程式及檔案

IDLE

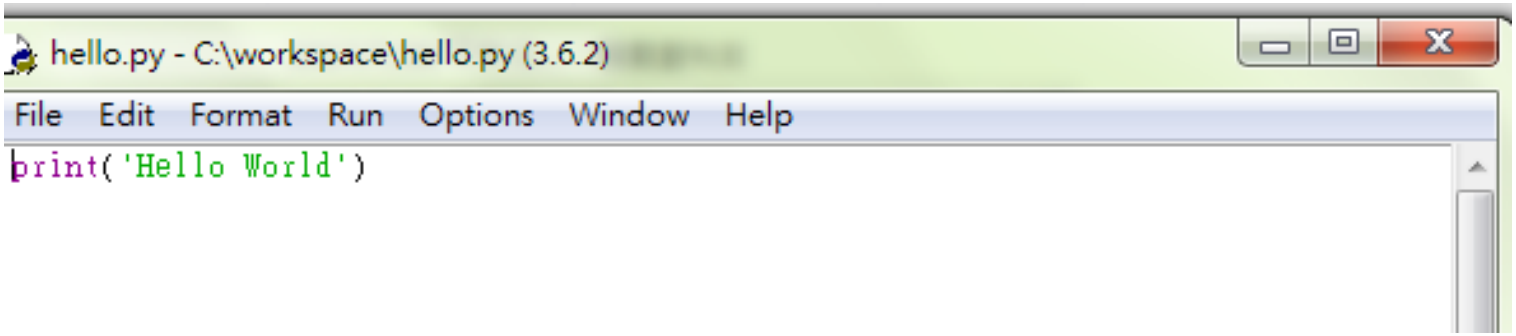

Python 3.6.2 Shell
File Edit Shell Debug Options Window Help
Python 3.6.2 (v3.6.2:5fd33b5, Jul 8 2017, 04:14:34) [MSC v.1900 32 bit (Intel)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>>
Ln: 3 Col: 4

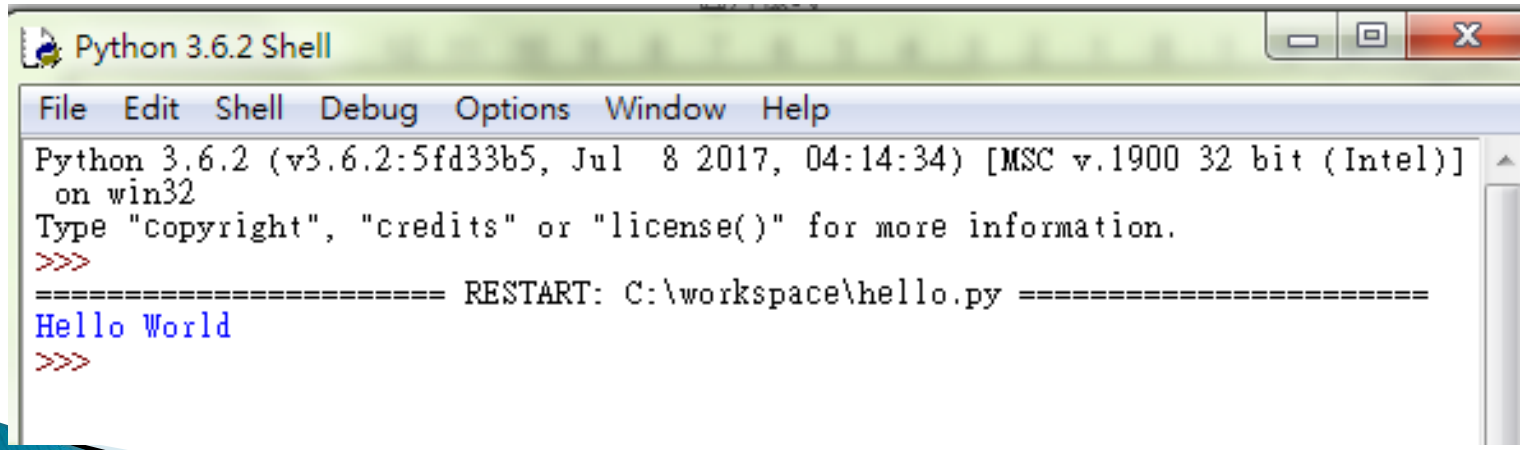
✓選取File->New File（如圖）來建立原始碼（source code）檔案。



執行python程式 – 編寫程式檔

- 編寫程式碼後，選取選單Run–> Run Module(按F5)來執行原始碼執行

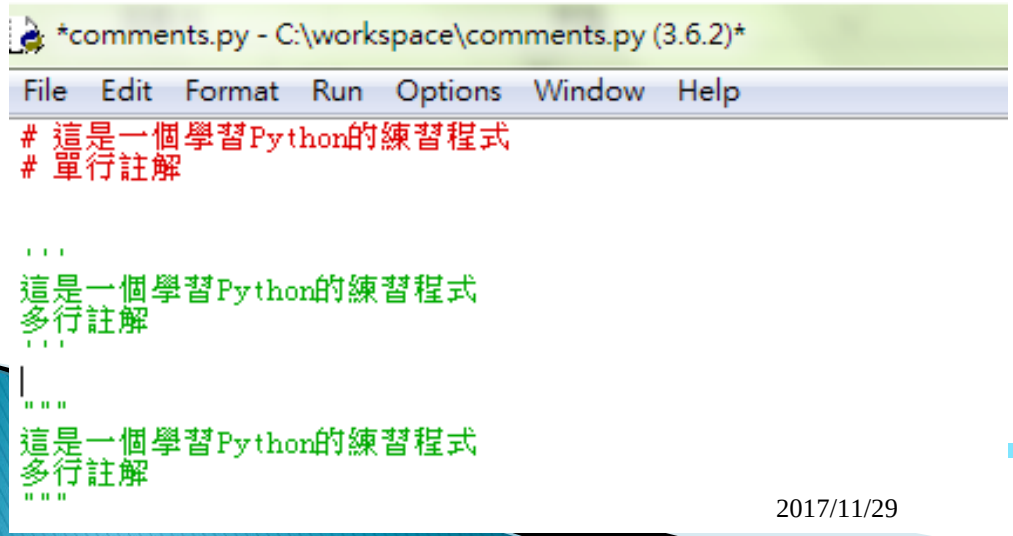




python的註解

- ✓單行註解：#之後到換行為止都是註解（所以最多註解單行）
- ✓多行註解：利用多行”””...”””

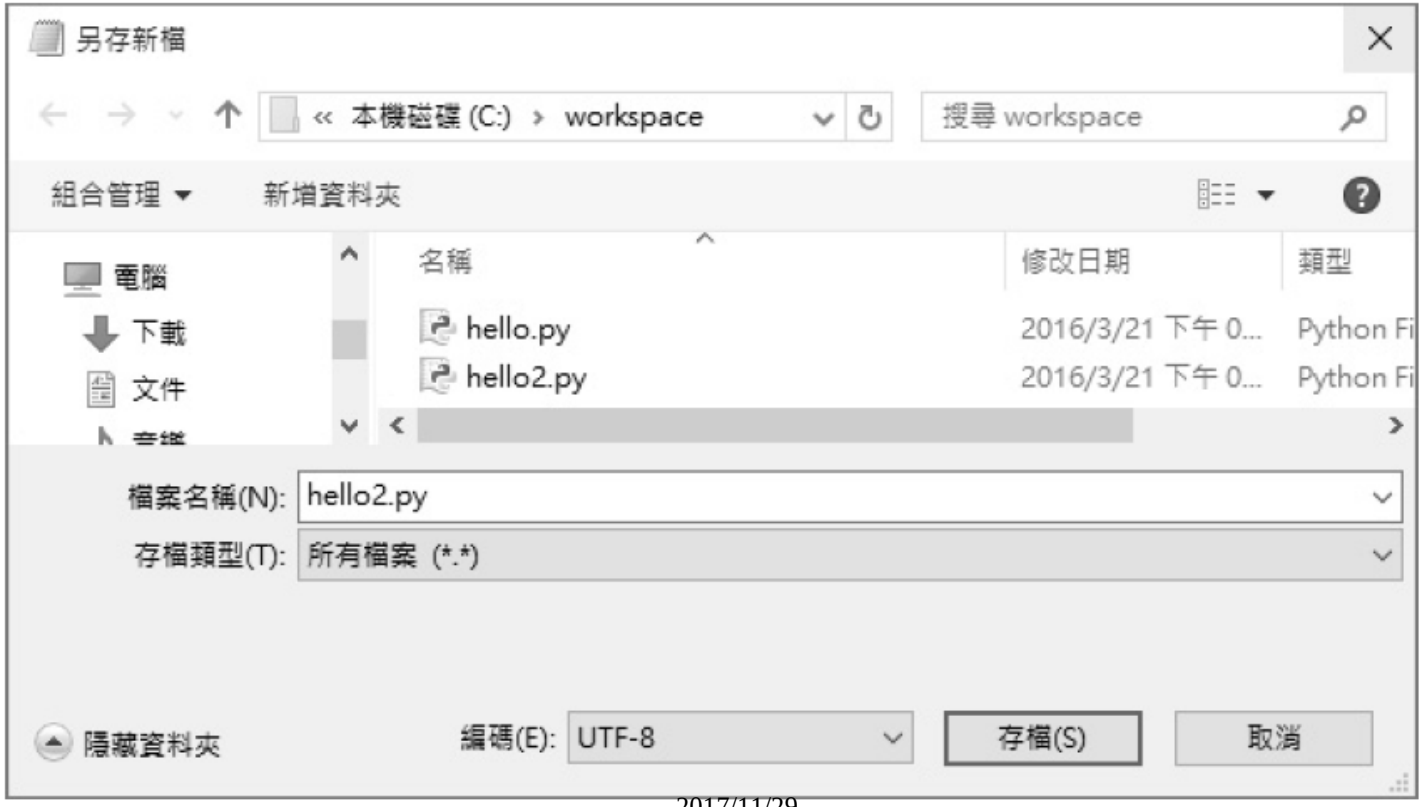
  ’’’...’’’
- ✓註解敘述可幫助程式設計師彼此溝通與了解程式。因為它不是程式敘述，所以直譯器不會理會註解敘述。



comments.py

UTF-8

✓Python 3 之後，python 直譯器預期的原始碼檔案編碼必須是 UTF-8



2-4 程式設計的錯誤

✓程式設計錯誤可分為三類：語法錯誤（syntax errors）、執行期間的錯誤（runtime errors），以及邏輯錯誤（logic errors）。

2-4-1　語法錯誤

✓一般最常見的錯誤是語法錯誤。如其他程式語言，Python有它自己的語法，因此撰寫程式碼時需遵守程式規則。若違反此規則，例如忘了加雙引號或字拼錯了，Python將產生錯誤訊息。

- ✓語法錯誤是來自於程式碼建構上的錯誤，比方說誤打敘述、不正確的內縮、遺漏了必要的符號，或是使用了起始大括號，卻漏掉相對應的終止大括號。

- ✓這類的錯誤通常很容易找到，因為Python會告訴您錯誤的地方以及造成錯誤的原因。比方說，下列的print敘述將會有語法錯誤。

  >>> print(Programming is fun')
  SyntaxError: invalid syntax

- ✓字串Programming is fun少了一個雙引號或是單引號。

2-4-2　執行期間的錯誤

- ✓執行期間的錯誤（Runtime errors）是會導致程式不正常終止的錯誤。
- ✓在程式執行時，如果執行環境偵測到無法進行的動作，便會出現Runtime Errors。例如，在整數除法運算中，當除數為零的時候將會出現執行期的錯誤。
- ✓下列程式碼的 1 / 0 運算式就會產生執行期間的錯誤。
- ✓數學運算的加、減、乘、除於Python的寫法是 + 、- 、* 、/ 。

```
>>> print(1/0)
Traceback (most recent call last):
  File "<pyshell#12>", line 1, in <module>
    print(1/0)
ZeroDivisionError: division by zero
```

2-4-3　邏輯錯誤

✓邏輯錯誤（logic errors）出現於程式執行的結果與預期中的不同。

基本概念

✓語法特色

- 以冒號(:)做為敘述的開始
- 不必使用分號(;)做為結尾
- 井字號(#)做為註解符號，同行井字號後的任何字將被忽略
- 使用tab鍵做為縮排區塊的依據
- 不必指定變數型態 (runtime時才會進行binding)

3. 型態與運算子

在Python裡，什麼都是物件

- ✓Python都是物件，或說是把任何資料包裝成物件，以型 別區分
- ✓其他語言，如C++與Java，有分一般變數、指標（指位 器）、參考（reference）等

物件皆有型別（type）

```
>>> 3
3
>>> 1.25
1.25
 >>> 'hello‘
 'hello‘
```

✓型別int（整數）、float（浮點數）、str（字串）

Python的內建型態

- ✓數值型態（Numeric type）
  - int, long, float, bool, complex
- ✓字串型態（String type）
- ✓容器型態（Container type）
  - list, set, dict, tuple

基本型態 (Numbers and String)

✓Numbers (數值)

```
a = 3                    # Integer (整數)
b = 4.5                  # Float point (浮點數)
c = 51728888333L         # Long Integer (精準度無限)
d = 4 + 3j               # Complex number (複數)
```

數學當中的複數，我們是用「i」來表示虛數的部份，而在Python當中則是用「j」來表示。

✓Strings (字串)

```
a = 'Hello'              # Single quotes
b = "World"              # Double quotes
```

數值型態

- ✓整數
  - ➢型態為 int，不再區分整數與長整數
  - ➢整數的長度不受限制（除了硬體上的限制之外）

```
>>> 10
10
>>> 0b1010
10
>>> 0o12
10
>>> 0xA
10
>>>
```

✓想知道某個資料的型態，可以使用 type()

```
>>> type(10)
<class 'int'>
>>> type(0b1010)
<class 'int'>
>>> type(0o12)
<class 'int'>
>>> type(0xA)
<class 'int'>
>>>
```

✓浮點數

- ➢float 型態

```
>>> type(3.14)
<class 'float'>
>>> 3.14e-10
3.14e-10
>>> float('1.414')
1.414
>>>
```

- ✓布林
  - ➢bool 型態
  - ➢只有 True 與 False 兩個值
  - ➢bool() 可將 0 轉換為 False，而非 0 值轉換為 True
  - ➢將 None、False、0、0.0、0j（複數）、‘’（空字串）、()（空 Tuple）、[]（空清單）、{}（空字典）等傳給 bool()，都會傳回 False，這些型態的其他值傳入bool() 則都會傳回 True

✓複數

- 型態為 complex
- 撰寫時使用a + bj 的形式

```
>>> a = 3 + 2j
>>> b = 5 + 3j
>>> a + b
(8+5j)
>>> type(a)
<class 'complex'>
>>>
```

字串型態

- ✓可以使用''或""包括文字
- ✓多數 Python 開發者的習慣是使用單引號

```
>>>
>>> 'abc'
'abc'
>>> "QQQ"
'QQQ'
>>>
```

命名規則

- ✓可使用英文字母小寫a到z、大寫A到Z、數字0 到9、底線「_」
- ✓第一個字不能是數字
- ✓區分大小寫，makewish與makeWish代表不同 的名稱
- ✓保留字（關鍵字）：Python已賦予特定意義
- ✓命名慣例

Lab 1

✓請實作練習下列的Python變數名稱，試看看有哪幾個是合法的。

(a) 94a
(b) _abc
(c) ab$c
(d) %abc
(e) abc4
(f) kk_cc
(g) cc-dd
(h) print

2-2 格式化輸出

- ✓若要將輸出結果看起來更美觀的話，則需借助格式化的輸出。格式化的輸出有三種格式，第一種是利用format函式。以下將配合範例說明之。
- ✓**1. format**的語法如下：
  - ➢print(format(item, format-specifier))
- ✓其中item表示數字或字串，format-specifier則為格式指定器。其中格式指定器是以字串的方式表示的，如表2-2所示：

表2-2　常用的格式指定器

格式指定器	說明
"10.2f"	以欄位寬10、準確度為2將浮點數加以格式化。
"10.2e"	以欄位寬10、準確度為2將浮點數以科學記號加以格式化。
"8d"	以欄位寬8及十進位將整數加以格式化。
"8x"	以欄位寬8及十六進位將整數加以格式化。
"8o"	以欄位寬8及八進位將整數加以格式化。
"8b"	以欄位寬8及二進位將整數加以格式化。
"10.2%"	將整數以百分比加以格式化，並以欄位寬10，準確度為2表示。
"30s"	以欄位寬30將字串加以格式化。
"<10.2f"	將浮點數格式化的項目向左靠齊。
">10.2f"	將浮點數格式化的項目向右靠齊。

- print(format(item, format-specifier))

- ✓當整數時，則使用 'd' 指定器；若為浮點數，則使用 'f'指定器；若為字串，則使用 's' 指定器。
- ✓而每一指定器前可加入數值，表示其欄位寬，如 '8d' 表示有8個欄位空間。還有向左或向右靠齊，分別以 < 和 > 來加以控制，它可以用於整數、浮點數，以及字串皆可。
- ✓表2-2的數字是為了配合說明而設定的，您可以視自己的需要而定。

➢print(format(item, format-specifier))

```
>>> print(format(100, '8d'))
       100
```

✓在100的前面應該會有5個空白，因為欄位寬為8，表示有8個空間，但100只有3位數，所以前面會空出5個空白。

➢print(format(item, format-specifier))

✓若指定的欄位寬小於要印出項目的空間時，此時欄位寬將被忽略之。

```
>>> print(format(100000, '3d'))
100000
```

✓因為印出100000至少要6個空間，但我們只給3位，所以此時的3將會忽略不用，而變為d的格式指定器而已。

➢print(format(item, format-specifier))

格式化字串

✓print() 函式的顯示預設是會換行

name = 'just in'

print('Hello')

print(name)

print() 有個 end 參數，在指定的字串顯示之後，end 參數指定的字串就會輸出

```
name = 'Justin'
print('Hello ', end = '')
print(name)
```

✓預設的分隔符號是一個空白字元，如果想要指定其他字元的話，可以指定 sep 參數

```
name = 'Justin'
print('Hello', name, sep = ', ') # 顯示 Hello, Justin
```

p3-1.py p3-2.py p3-3.py

格式化輸出

✓練習題

- p2-1.py
- p2-2.py
- p2-3.py