06 物件導向

黃彬華編撰

- 程式語言為什麼離不開物件導向？
- 文件字串
- 傳值與傳參照有什麼差別？
- 兩個值相同的物件，位址會相同嗎？
- 類別變數與實例變數有什麼差別？
- 不同種類方法該怎麼用？
- 一大堆物件要如何處理？
- 子類別可以繼承到什麼財產？
- 父類別方法內容不適用可否改掉？
- 多重繼承會不會更加富有？
- 聽說型別檢查有點麻煩
- 多型可以解決型別檢查的麻煩嗎？
- 抽象類別很抽象嗎？
- 何謂值相同？
- 搜尋與排序很常用，你會了嗎？
- 為什麼要封裝？
- Package, Module傻傻分不清？

程式語言為什麼離不開物件導向？ - 1

黃彬華編撰

- 如果以傳統方式來記錄書籍資料，最直覺的方式就是建立表格來儲存
- 物件導向的觀念，就是將1本書以1個物件來表示，也同樣要將該物件所儲存的資訊填到表格內，只不過這張表格是放在記憶體裡

書		
書名	定價	作者
Python	500.0	Paul
iOS App	600.0	Ivy

程式語言為什麼離不開物件導向？ - 2

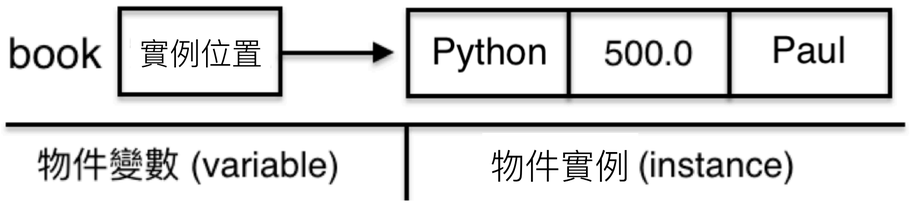
黃彬華編撰

- 建立一筆資料的順序與建立物件的順序其實是一致的
- 建立表格
  - 定義標題與欄位
  - 建立一筆資料以記錄一本書籍資訊
- 建立物件
  - 定義類別與屬性
  - 建立一個物件以儲存一本書籍資訊

程式語言為什麼離不開物件導向？ - 3

黃彬華編撰

- 定義類別 (class)
  - class Book:
- 定義屬性與 __init__() 方法
  - 屬性 (attribute)，又稱實例變數 (instance variable)
  - __init__() 方法用於實例化過程
  - 呼叫__init__()方法會建立物件實例 (instance)
    - book = Book("Python", 500, "Paul")



程式語言為什麼離不開物件導向？ - 4

黃彬華編撰

- 一個類別只能定義一個 __init__()方法 (建構式)，專供產生物件實例時呼叫
  - 只能定義一個是因為方法參數可以設定預設值，所以無需overloading
- __init__() 方法雖然很像一般方法，但是有3個不同點
  - __init__() 方法有固定名稱，一般方法則無
    - __init__() 方法名稱必須為「 __init__() 」
    - 一般方法可隨意取名，只要符合識別字命名規則即可
  - __init__() 方法無回傳值，一般方法可有可無
    - __init__() 方法主要目的在設定屬性初始值，所以不需要回傳值
  - 呼叫時機不同
    - 產生物件實例需要呼叫 __init__() 方法，而且一個實例只會呼叫一次
    - 物件實例產生後可任意、多次呼叫一般方法

程式語言為什麼離不開物件導向？ - 5

黃彬華編撰

- 類別內的函式稱作方法
  - 實例方法第一個參數需要為self，以接收傳來的物件，方便使用物件內的屬性
- 透過呼叫相同的方法以達到運算式的重複利用
  - 定義

```
def show(self):
    print(f"name: {self.name}")
    print(f"price: {self.price}")
    print(f"author: {self.author}")
```

  - 呼叫

```
book1.show()
book2.show()
.....
```

範例

- ClassObjectDemo

文件字串

- 可以為類別、函式加上文件字串 (docstrings)
  - 類別或函式內第一行以三引號 (''' 或 """) 加上文件字串
  - 游標滑到該類別、函式上會跳出說明文字
  - 「類別名.__doc__」 或「函式名.__doc__」可取得對應文件字串

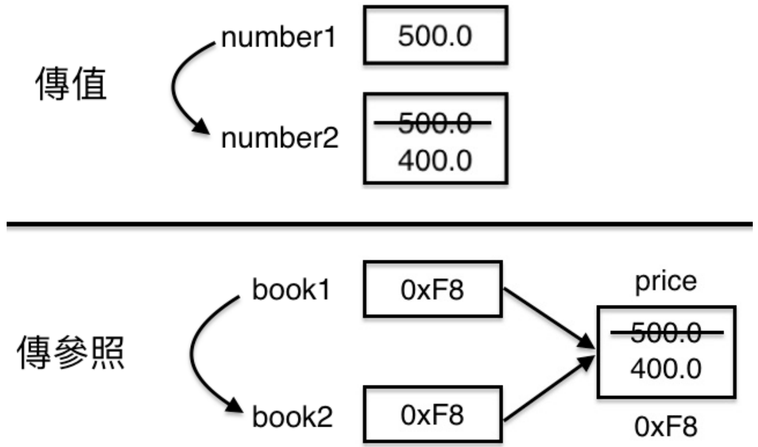
範例

- DocstringDemo

傳值與傳參照有什麼差別？

黃彬華編撰

- 傳值 (pass by value)
  - 以複製值的方式做指派
  - int, float, bool等基本類型屬之
- 傳參照 (pass by reference)
  - 以複製位置方式做指派
  - 物件指派時屬於傳參照



範例

- PassByDemo

練習 6-1

黃彬華編撰

- 建立 1 個長方體 (Cuboid) 類別，內容包含
  - 屬性：長 (length)、寬 (width)、高 (height)
  - 方法
    - volume() 方法：計算完體積並回傳
    - getInfo() 方法：回傳長、寬、高與體積
  - 建構式：設定屬性初始值
- 主流程
  - 將使用者輸入的長、寬、高建立 Cuboid 物件，並顯示該物件長、寬、高與體積

```
請輸入長方體的長、寬、高(空白間隔): 1 2 3
輸入的長方體資訊如下:
長: 1.0, 寬: 2.0, 高: 3.0, 體積: 6.0
```

兩個值相同的物件，位址會相同嗎？

黃彬華編撰

- 比較運算符號的 == 是比較值是否相同
- 相等運算符號 (identity operator)，是比較儲存的位址是否相同
  - is
  - is not

範例

- ObjectComparisonDemo

類別變數與實例變數有什麼差別？

黃彬華編撰

- 類別變數 (class variable) 與實例變數 (instance variable)比較
  - 變數宣告點
    - 類別變數宣告在類別內，但不在其他方法內
    - 實例變數宣告在__init__()方法參數上
  - 變數分享範圍
    - 相同類型物件可存取類別變數
    - 物件實例只能存取自己的實例變數
  - 搭配取值方法
    - 類別方法 (class method) 存取類別變數
    - 實例方法 (instance method) 存取實例變數
  - 存取方式
    - 用類別來存取類別變數
    - 用物件來存取實例變數

範例

- ClassVariableDemo

不同種類方法該怎麼用？

- 實例方法 (instance method)
  - 以物件呼叫，第一個參數會傳入該物件，方便存取該物件的實例變數
- 類別方法 (class method)
  - 需加上@classmethod
  - 以類別呼叫，第一個參數會傳入該類別，方便存取該類別的類別變數
- 靜態方法 (static method)
  - 需加上 @staticmethod
  - 以類別呼叫，但不會將該呼叫的類別傳入，所以定義參數跟呼叫的類別不需要有任何關係

範例

- InsClaStaMethodDemo

一大堆物件要如何處理？

- list可以儲存大量物件，並可套用下列常用函式
  - map
  - filter
  - max
  - min
  - reduce

範例

- ObjectListDemo

練習 6-2

黃彬華編撰

- 承襲 6-1，不過改成可以輸入多個長方體資訊，所以 Cuboid 類別新增一個static方法
  - getCuboidsInfo(cuboid)：回傳所有長方體資訊
- 主流程
  - 詢問使用者欲輸入的長方體總個數
  - 讓使用者輸入指定個數的長方體資訊，並顯示所有輸入長方體的長、寬、高與體積

```
請問有幾個長方體? 3
請輸入第1個長方體的長、寬、高(空白間隔): 1 1 1
請輸入第2個長方體的長、寬、高(空白間隔): 2 2 2
請輸入第3個長方體的長、寬、高(空白間隔): 3 3 3
輸入的3個長方體資訊如下:
長: 1.0, 寬: 1.0, 高: 1.0, 體積: 1.0
長: 2.0, 寬: 2.0, 高: 2.0, 體積: 8.0
長: 3.0, 寬: 3.0, 高: 3.0, 體積: 27.0
```

子類別可以繼承到什麼財產？

黃彬華編撰

- 透過繼承 (inheritance) ，子類別 (derived class) 可以重複利用父類別 (base class) 定義的內容
  - 定義父類別 (base class) 與實例變數

```
class Book:
    def __init__(self, name="", price=0.0, author=""):
```

  - 定義子類別 (derived class) 與實例變數，但在子類別的 __init__() 方法內必須先呼叫父類別對應方法

```
class ComputerBook(Book):
    def __init__(self, name="", price=0.0, author="", exampleUrl=""):
        super().__init__(name, price, author)  # 呼叫父類別 __init__()  方法
        self.exampleUrl = exampleUrl
```

- self 與 super 關鍵字
  - self 用來存取所在類別的成員
  - super 則可存取父類別的成員

範例

- InheritDemo

父類別方法內容不適用可否改掉？

黃彬華編撰

- 如果父類別方法內容不適用於子類別，子類別可以改變內容以符合需求，稱作覆寫 (override) ，規則如下：
  - 方法名稱要一致
  - 參數個數要一致

範例

黃彬華編撰

- OverrideDemo

練習 6-3

黃彬華編撰

- 承襲 6-2，但再建立 1 個長方體類別的子類別：房屋 (House) 類別，內容包含
  - 屬性：增加材質 (material) 屬性
  - 方法：覆寫getInfo()方法：除了回傳長、寬、高與體積外，還能回傳材質
  - 建構式：設定長、寬、高、材質等 4 屬性的初始值
- 主流程
  - 詢問使用者欲輸入的房屋總個數
  - 讓使用者輸入指定個數的房屋資訊，並顯示所有輸入房屋的長、寬、高與體積

```
請問有幾間房屋? 3
請輸入第1間房屋的長、寬、高與材質(空白間隔): 1 1 1 C
請輸入第2間房屋的長、寬、高與材質(空白間隔): 2 2 2 B
請輸入第3間房屋的長、寬、高與材質(空白間隔): 3 3 3 A
輸入的3間房屋資訊如下:
長: 1.0, 寬: 1.0, 高: 1.0, 體積: 1.0, 材質: C
長: 2.0, 寬: 2.0, 高: 2.0, 體積: 8.0, 材質: B
長: 3.0, 寬: 3.0, 高: 3.0, 體積: 27.0, 材質: A
```

根類別

黃彬華編撰

- object 類別 是所有 Python 類別的父類別，也就是根類別 (root class)
- object 類別定義的方法
  - 可被其他類別覆寫
  - 可被其他物件呼叫

範例

- RootClassDemo

多重繼承會不會更加富有？

黃彬華編撰

- Python 支援多重繼承

  class ProgramBook(ComputerBook):

  class DatabaseBook(ComputerBook):

  class PythonDBBook(**ProgramBook, DatabaseBook**):

- 多個父類別定義相同方法時，子類別會呼叫最左邊，也就是第一順位父類別的方法

範例

- MultiInheritDemo.py

聽說型別檢查有點麻煩

黃彬華編撰

- 宣告一個可以處理任何型別的list
  - cart = [ ]
  - 例如購物車
- 從 list 取物件出來必須
  - 使用「isinstance」檢查型別，方能呼叫該型別定義的方法
  - 型別眾多時判斷上很麻煩

範例

- TypeCheckDemo

多型可以解決型別檢查的麻煩嗎？

黃彬華編撰

- 多型 (polymorphism)
  - 定義一個所有物件的共同父類型
  - 建立各個子類型並覆寫父類型的方法。產生各個子類型物件實例並儲存至集合內
  - 執行時會依據實例類型呼叫對應的覆寫方法

範例

- PolymorphismDemo

抽象類別很抽象嗎？

黃彬華編撰

- 抽象類別 (abstract class) 可以說是藍圖、規範，用以約束各個子類別要實作必要功能
  - 如果不訂規範就沒有約束力，各個開發者各自為政，增加呼叫者麻煩
  - 舉 DAO 為例：約束各個 DAO 開發者遵守規範，寫出相同功能方便呼叫者使用
- 抽象類別建立步驟如下
  - 匯入 abc module 所需功能
    - from abc import abstractmethod, ABC, ABCMeta
  - 透過「metaclass=ABCMeta」或繼承ABC類別建立抽象類別
  - 抽象類別內定義抽象方法 (abstract method)；抽象方法只定義架構，沒有內容
    - 加上「@abstractmethod」標示為抽象方法，抽象方法內容直接以 pass 帶過
- 抽象類別內有抽象方法則無法實例化 (沒有抽象方法則可實例化)
- 子類別需要實作 (implement) 抽象父類別的所有抽象方法，否則該子類別也無法實例化

範例

黃彬華編撰

- AbstractDemo
- DaoDemo

何謂值相同？

- 欲加入的元素不可與set內既存的元素值相同，否則無法加入，但何謂「值相同」？
  - 元素所屬類別必須覆寫 object.__eq__() 與 object.__hash__() 方法
    - object.__eq__()，只能實現循序搜尋 (sequential search)
    - object.__hash__()，可以實現雜湊搜尋 (hash search)
  - object 類別是所有類別的父類別
    - object 類別說明參看 class object
    - object 類別的方法參看 Basic customization

範例

- SetHashValueDemo

搜尋與排序很常用，你會了嗎？

黃彬華編撰

- 搜尋
  - 在 list 內搜尋自訂物件需要覆寫 __eq__()，但無須覆寫 __hash__()
- 排序 (以下說明適用於 list 與 set)
  - 可以使用 lambda 指定要比對的屬性，預設為升冪，reverse=True則為降冪
    - sortedList = sorted(myBooks, key=lambda book: book.price, reverse=True)
  - 也可以引用 operator 的 attrgetter() 指定要比對的屬性
    - from operator import attrgetter
    - sortedList = sorted(myBooks, key=attrgetter("price", "name"))

範例

黃彬華編撰

- SearchSortDemo

練習 6-4

黃彬華編撰

- 站名: 台中
- 緯度: 24.111111
- 經度: 121.111111
- 站名: 台中, 緯度: 24.111111, 經度: 121.111111
- 繼續輸入 (Y | y): y
- 站名: 台中
- 緯度: 24.111111
- 經度: 121.111111
- 台中站已經存在
- 站名: 台北
- 緯度: 24.333333
- 經度: 121.111111
- 站名: 台北, 緯度: 24.333333, 經度: 121.111111
- 繼續輸入 (Y | y): y
- 站名: 桃園
- 緯度: 24.222222
- 經度: 121.111111
- 站名: 桃園, 緯度: 24.222222, 經度: 121.111111
- 繼續輸入 (Y | y): n
- 車站依照緯度高到低排序如下:
- 站名: 台北, 緯度: 24.333333, 經度: 121.111111
- 站名: 桃園, 緯度: 24.222222, 經度: 121.111111
- 站名: 台中, 緯度: 24.111111, 經度: 121.111111

- 使用者可以新增多個車站的站名、緯度、經度資料
  - 站名不可重複
- 新增完畢後會依照緯度高到低排序後顯示車站資訊

為什麼要封裝？ - 1

- 要懂封裝 (encapsulation) 前，要先了解存取修飾詞 (access modifiers)，共有下列3種，開放程度高到低依序為
  - public
    - 屬性與方法預設為public，類別以外都能隨意存取而沒有限制
  - protected
    - 屬性與方法前要加上 "_" (1個底線)，子型可以存取
  - private
    - 屬性與方法前要加上 "__" (2個底線)，類別以外都不開放

為什麼要封裝？ - 2

- 函式庫開發者可將複雜度封裝 (encapsulation) 起來，也可避免外部不當存取屬性。封裝步驟如下：
  - 將屬性以 private 修飾詞封住，不讓外部直接存取
  - 建立 public setter，檢查傳入值是否正確以決定是否指派給 private 屬性
  - 建立 public getter，並回傳 private 屬性
  - 可以使用 "@property" 建立 getter；"@field_name.setter" 建立 setter

範例

- EncapsulationDemo

Package, Module傻傻分不清？

黃彬華編撰

- 一個 module 就是一個 Python 檔案
- 一個 package 就是一個目錄，所以可以有sub package，也就是子目錄
  - package 內有多個 module
- from [module] import [class]
  - 使用同一個 module 內資源不需要使用 from-import
  - 使用相同 package 但不同 module，可以省略 package 名稱
    - from .[module] import [class]
  - 使用不同package
    - from [package].[module] import [class]
- 要將欲執行的檔案 (例如 main.py) 視為 module，否則執行錯誤
  - python -m myPackage.packageC.main

範例

- myPackage
- 要將欲執行的檔案 (例如 main.py) 視為 module，否則執行錯誤
  - python -m myPackage.packageC.main