EXCEL VBA 進階班

Lecture 7

SECTION 11. 物件導向簡介

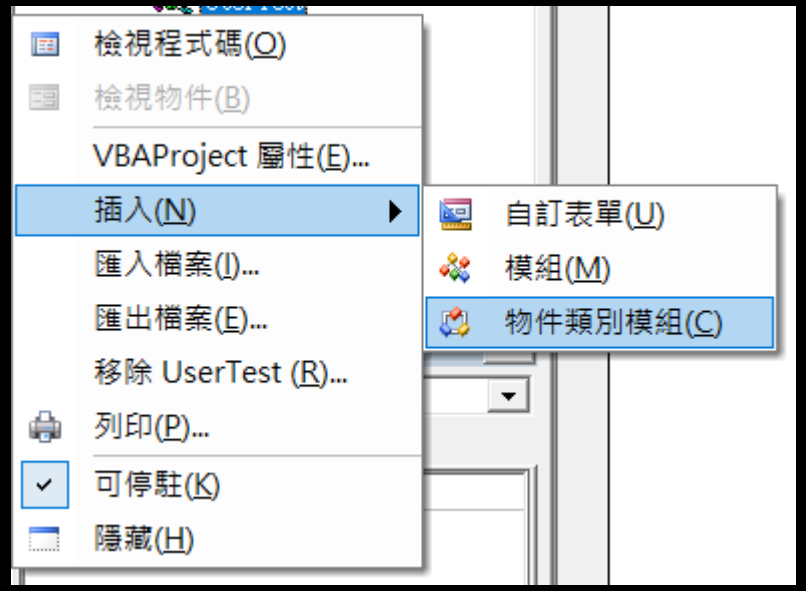
衍生性金融商品損益計算

物件導向簡介

- 回到 Lecture 6 的問題，我們要怎麼建立我們的類別，讓程式碼變得更容易維護呢？
- 假如我們現在要面對的問題不只是買權，還有賣權，那我們需要的是什麼？
- 首先我們要創造一個選擇權的類別，把它命名為 PlainVanilla，類別的命名與函數或是子程序不同，開頭習慣使用大寫。

物件導向簡介：創造類別

- 在專案處按下滑鼠右鍵 ➔ 選擇插入 ➔ 選擇物件類別模組。

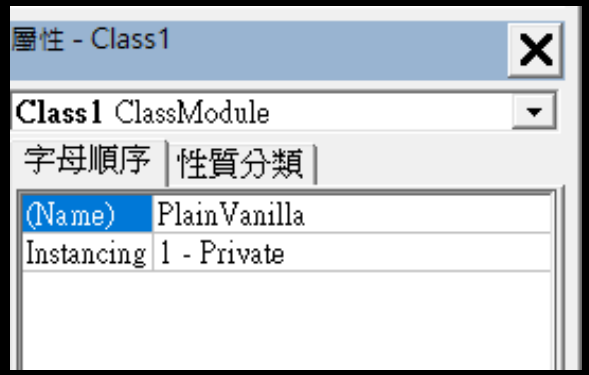


物件導向簡介：創造類別

- 接著會看到下方出現了一個物件類別模組，裡面有一個 Class1

物件導向簡介：創造類別

- 該 Class1 就是我們新創立的類別，將它命名為 PlainVanilla，修改屬性 (Name) 處即可。



物件導向簡介：創造類別

- 接著我們可以在其他的模組中宣告一個 PlainVanilla 試試看了。

```
Sub test()

Dim financialInstrument As New PlainVanilla

End Sub
```

- 我們宣告了 financialInstrument 是一個新的 PlainVanilla 類別物件。

物件導向簡介：創造類別

- 如果是要宣告一個新的物件，New 這個關鍵字是不可或缺的，除了上述方法，也可以利用：

```
Sub test()

Dim financialInstrument As PlainVanilla
Set financialInstrument = New PlainVanilla

End Sub
```

物件導向簡介：創造類別

- 為何一定要 New ?因為如果不使用 New，則 VBA 不會再記憶體理新增空間給一個新的 PlainVanilla 物件，當我們使用以下程式碼：

```
Sub test()

Dim firstInstrument As New PlainVanilla
Dim secondInstrument As PlainVanilla
Set secondInstrument = firstInstrument

End Sub
```

- 其實是代表 firstInstrument 與 secondInstrument 指向同一個記憶體位址。

物件導向簡介：創造類別

- 上面情況造成 secondInstrument 其實只是 firstInstrument 的別名，改動一者另外一者會跟著更動。
- 稍後介紹屬性時更能看出該點。

物件導向簡介：屬性

- 創造了一個選擇權的類別後，我們需要賦予他各種選擇權的特性。
- 歐式選擇權擁有：
  - 履約價：一個大於零的約定價格，讓我們到時候可以以該價格買賣某個商品。
  - 到期日：該天可以決定是否履約。
  - 選擇權型態：是買權或賣權。
- 上述三者為選擇權內所存有的資訊，稱為選擇權的屬性（Property）。

物件導向簡介：屬性

- 因此我們在選擇權中加入屬性，加入方式與加入全域變數的方式相同，在整個類別模組上面加入三個變數即可。
- 由於選擇權不是買權就是賣權，為了方便記錄我們使用 enumerator。

```
Option Explicit
Option Base 1

Public strike As Double
Public expiryDate As Date
Public putCallType As OptionType
```

```
Public Enum OptionType
    callOption = 0
    putOption = 1
End Enum
```

物件導向簡介：屬性

- 現在宣告完變數後，我們可以在裡面填入需要的資料。

```
Sub test()

Dim firstInstrument As New PlainVanilla
firstInstrument.strike = 100
firstInstrument.expiryDate = DateValue("2013/01/02")
firstInstrument.putCallType = callOption

End Sub
```

物件導向簡介：屬性

- 從這裡我們可以來試試看沒有使用 New 所造成的問題，下面情況 seconedInstrument 的 strike 會變為多少？

```
Sub test()

Dim firstInstrument As New PlainVanilla
firstInstrument.strike = 100
Dim seconedInstrument As PlainVanilla
Set seconedInstrument = firstInstrument
firstInstrument.strike = 999
MsgBox seconedInstrument.strike '會是多少？

End Sub
```

物件導向簡介：屬性

- 因此這也造成一個問題，當我們需要一個新物件，只與某一個已經存在的物件相差一小部分時，我們不能利用 Set 新物件 = 已存在物件 的方式。
- 該如何解決這個問題？之後在方法的部分會再詳加敘述。

物件導向簡介：屬性

- 到目前為止，物件導向跟 user-defined type 相較起來完全相同，並沒有任何優勢。
- 現在我們要介紹他的一大優勢，不同於 Sub 與 Function，類別還存在 Property。
- user-defined type 最讓人詬病的地方在於，不存在一個機制來檢查傳入的值是否符合我們想要的型態。
- 例如之前的 TimeSeries，我們沒有辦法來檢查日期與值的長度是否相同，即使不同程式還是可以運作，造成除錯上的困難。

物件導向簡介：PROPERTY

- 現在回到 PlainVanilla，一個選擇權的履約價，strike，其實應該大於零，但我們現在將 strike 設為 Public，等於可以任意修改，即使傳入小於等於零的數字也不會發生錯誤。
- 因此我們應該有一個機制，讓我們每次輸入 strike 時可以先行檢查是否合理。
- 首先，先將 strike 改為 private，會看到我們無法在測試的子程序中修改 strike 了。

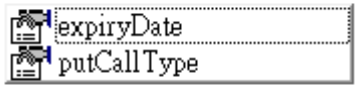
物件導向簡介：PROPERTY

```
Option Explicit
Option Base 1

Private strike As Double
Public expiryDate As Date
Public putCallType As OptionType
```

```
Sub test()

Dim instrument As New PlainVanilla
instrument.
End Sub
```



物件導向簡介：PROPERTY

- 但我們還是需要有一個方法將履約價輸入至 strike 屬性之中。
- 因此這時候，
  - 我們將 strike 更名為 strike_。
  - 建立一個 Property 名為 strike。

物件導向簡介：PROPERTY

```
Option Explicit
Option Base 1

Private strike_ As Double
Public expiryDate As Date
Public putCallType As OptionType

Property Let strike(ByVal Value As Double)

End Property
```

物件導向簡介：PROPERTY

- 這時候我們真正的履約價為存在 PlainVanilla 中的 strike_。
- 而透過 Property，我們可以用一個名為 strike 的子程序來檢測我們存取的結果是否正確無誤。
  - Let : 將值傳入。
  - Set : 將物件傳入。
  - Get : 將值輸出。

物件導向簡介：PROPERTY

- Property Let strike，幫助我們檢查傳入的履約價是否為一個合理的數字，並將該值傳遞給 strike_

```
Public Property Let strike(ByVal Value As Double)

If Value <= 0 Then
    Err.Raise Number:=10001, Source:="strike", Description:="履約價需大於0"
End If

strike_ = Value
End Property
```

物件導向簡介：PROPERTY

- Property Let 子程序最右邊的引數，是代表實際使用該子程序時等號右邊的輸入值。
- 我們將履約價輸入看起來與之前相同，但實際上是執行 Let strike 子程序。

```
Sub test()

Dim instrument As New PlainVanilla
instrument.strike = 100

End Sub
```

物件導向簡介：PROPERTY

- 可以試試看給 instrument 一個負的履約價：

```
Sub test()

Dim instrument As New PlainVanilla
instrument.strike = -100

End Sub
```

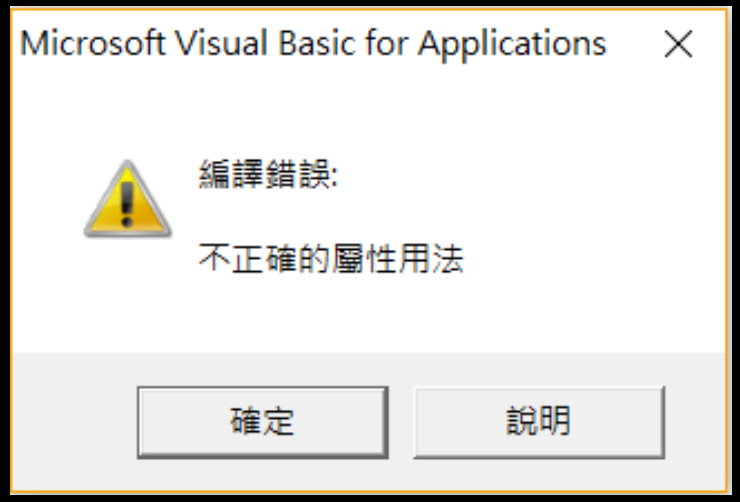
物件導向簡介：PROPERTY

- 但反過來會發現，我們沒有方法將 strike_ 的值傳出 PlainVanilla

```
Sub test()

Dim instrument As New PlainVanilla
Dim x As Double
instrument.strike = 100
x = instrument.strike

End Sub
```



物件導向簡介：PROPERTY

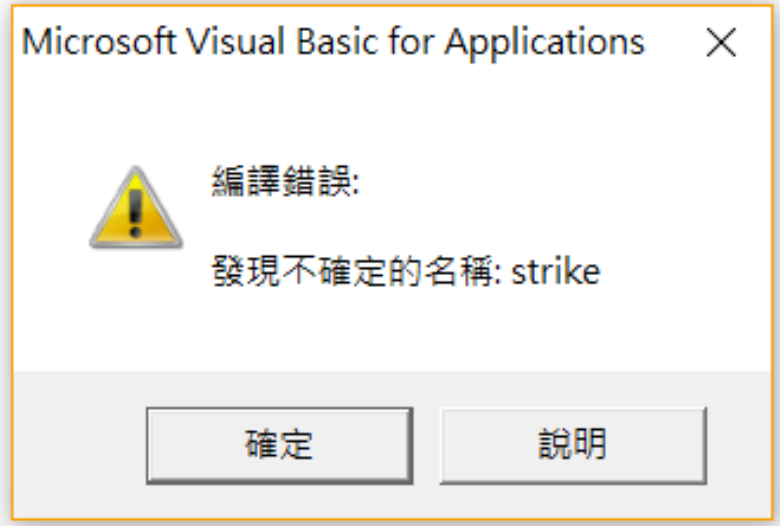
- 因此我們還需要一個函數，將 strike_ 的值傳出，如果能與 strike 使用相同名稱就太好了，看起來就像 PlainVanilla 中真的有一個變數叫 strike 一樣，使用上不會混淆很方便。

```
Public Function strike() As Double
strike = strike_
End Function
```

物件導向簡介：PROPERTY

- 但很遺憾，函數與子程序名稱必須唯一，在類別中也是一樣：

```
Public Function strike() As Double
strike = strike_
End Function
```



物件導向簡介：PROPERTY

- 因此我們不能用一般的函數，而是同樣使用 Property，只是中間部分改為 Get：

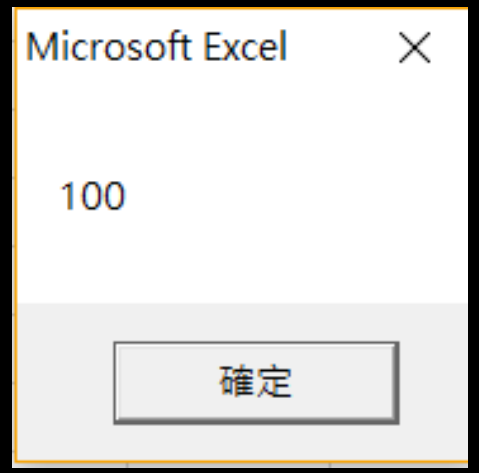
```
Public Property Get strike() As Double
strike = strike_
End Property
```

物件導向簡介：PROPERTY

- 修改完後，可以再度使用 strike 將值傳出了。

```
Sub test()

Dim instrument As New PlainVanilla
Dim x As Double
instrument.strike = 100
x = instrument.strike
MsgBox x
End Sub
```



- 同上述問題，大家可以想一想我們為何要將履約價變數改為 strike_？

物件導向簡介：封裝

- 我們利用 Property 偽裝我們有一個屬性叫 strike，並利用一個子程序與一個函數來讓該屬性在等號左右邊都能成立。
- 到上面為止為基本的屬性設定，其中物件導向一件很重要的特性，封裝（ Encapsulation ），已經被我們介紹到。

物件導向簡介：封裝

- 我們不允許類別外部的其他程式任意更動履約價為一個不合理的數值，因此將它設為了 Private。
- 要改變履約價僅能透過兩個 strike 程式來更動，避免發生不必要的錯誤。
- 這就是封裝的概念，使得我們的程式碼更容易維護與增加安全性。

物件導向簡介：方法

- 持有一個歐式選擇權我們能做一件事：
  - 在到期日時依照當時現貨價決定履約與否。
- 要物件進行一個動作被稱為一個方法（Method）。
- 方法命名開頭為動詞，因所有方法皆為一個動作。
- 因此現在我們要讓 PlainVanilla 物件有一個方法為計算報酬，命名叫 calculatePayoff。

物件導向簡介：方法

- 方法使用單純的函數或子程序，無需使用 Property。

```
Public Function calculatePayoff(spot As Double) As Double
Select Case putCallType
    Case callOption
        calculatePayoff = IIf(spot > strike, spot - strike, 0)
    Case putOption
        calculatePayoff = IIf(spot < strike, strike - spot, 0)
End Select
End Function
```

物件導向簡介：練習

- iif 函數為 VBA 版的 IF 函數，用法與 Excel 中的 IF 函數完全相同。
- 上述函數計算上沒問題，但會造成一個困擾：
  - 每次執行都必須做一次判斷是買權或賣權，重複執行時會多出很多無謂的判斷。
- 要如何利用 Property 與修改屬性讓這個判斷消除？

物件導向簡介：方法

- 前面有說到，我需要複製一份現有物件並修改這件事情並沒有辦法利用等號傳值得到該結果。
- 因此，每個類別理都需要存在一個可以複製出該物件的方法，命名為 clone。
- 以 PlainVanilla 為例：

物件導向簡介：方法

```
Public Function clone() As PlainVanilla
Dim cloneOption As New PlainVanilla

cloneOption.strike = strike_
cloneOption.expiryDate = expiryDate
cloneOption.putCallType = putCallType

Set clone = cloneOption
End Function
```

物件導向簡介：方法

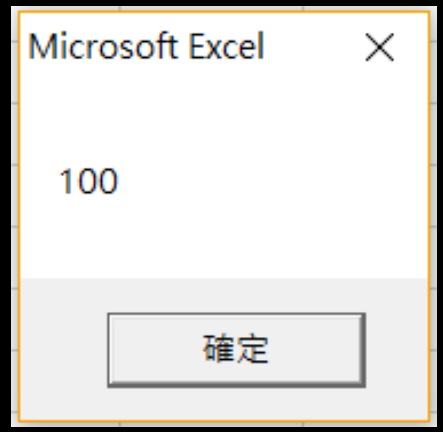
- 使用 clone 後則可複製出一份跟變化前一樣的 instrument1 :

```
Sub test()

Dim instrument1 As New PlainVanilla
instrument1.strike = 100

Dim instrument2 As New PlainVanilla
Set instrument2 = instrument1.clone
instrument1.strike = 1000
MsgBox instrument2.strike

End Sub
```



物件導向簡介：練習

- 歐式擇權其實只能在到期日當天決定履約與否，並得到報酬。
- 因此我們如果給定一個現貨商品價格的時間序列，我們會希望他回傳選擇權報酬之時間序列，思考一下如何完成以下函數：

```
Public Function calculatePayoff(spot As TimeSeries) As TimeSeries

End Function
```

物件導向簡介：CONSTRUCTOR

- 每次宣告物件我們都必須宣告完在一行一行輸入屬性，會降低程式的可讀性與安全性。
- 建立類別實體時，免不了會有初始化資料成員的動作。即使用在程式中設計了「Let XXX」等方法，但在使用上仍會感到不方便，尤其是需要初始化的資料成員相當 多時。較好的解決方案是建立物件時就初始化必要的資料成員。
- 許多物件導向程式語言因此有了建構子（Constructor）來幫助初始化物件，如下頁 C++ 範例，我們可以用語該類別同名之函數來做為一個建構子：

物件導向簡介 : CONSTRUCTOR

```
#include <Rcpp.h>
#include <string>
using namespace Rcpp;

enum OptionType{call,put}

class PlainVanilla{
public:
  PlainVanilla(double strike, Rcpp::Date expiryDate, OptionType type){

  }

};
```

物件導向簡介：CONSTRUCTOR

- 但遺憾 VBA 並沒有建構子，但我們可以自行新增一個函數來解決該問題。
- 我們令外創一個模組命名為 Constructors，用來放之後各個我們會用到的物件之建構子。

物件導向簡介：CONSTRUCTOR

```
Public Function newPlainVanilla(strike As Double, _
                                expiryDate As Date, _
                                putCallType As OptionType) As PlainVanilla

Dim newOption As New PlainVanilla
newOption.strike = strike
newOption.expiryDate = expiryDate
newOption.putCallType = putCallType

Set newPlainVanilla = newOption
End Function
```

物件導向簡介 : CONSTRUCTOR

- 之後我們即可透過建構子還產生一個新的物件。

```
Sub test()

Dim instrument As PlainVanilla
Set instrument = newPlainVanilla(100, DateValue("2011/2/3"), putOption)

End Sub
```

物件導向簡介 : CONSTRUCTOR

- 值得一提的是，利用該函數之後，我宣告物件時並不需要再使用 New
- 因為等號右邊的建構子每次都會回傳一個新的 PlainVanilla。
- 該建構子可以省去因為忘記使用 New 而造成的一些錯誤。

物件導向簡介：練習

- 將 Lecture 6 的範例新增選擇權為 Call 或 Put，程式改為使用 PlainVanilla 物件。
- 擴展我們的 PlainVanilla，現在買賣不再是只有一單位（例如是匯率選擇權的話則是 main amount 的數目不為 1），我們可以在裡面加入 amount 屬性，那此時損益計算會如何？
- 我們不只可以持有選擇權，也可以賣出選擇權，賣出當買方履約時，我們損益會如何？可以在裡面加入 buySellType 屬性作調整。