Chapter

02 基本語法

本章將介紹一些程式語言的基本語法，例如保留字、識別字的命名、資料型態、運算子及運算式等。

2-1 保留字與識別字

本單元先介紹保留字與識別字。

⊙ 保留字（Keywords）

保留字或稱關鍵字，是任一程式語言已事先賦予某一字元或字串一個特別意義，所以程式設計者不得再重複賦予不同的用途。例如：If 已被賦予決策敘述，程式設計者當然不得再定義 If 為另外的用途。就如同現實的社會，沒有人會將名字取為洪電視或林電腦。例如：以下是 VB 的一些常用保留字。

```
And As Boolean If For Short Integer + - * /
```

⊙ 識別字（Identifier）

真實的世界裏，每個人、事及物都有一個名稱，程式設計亦不例外。例如：於程式設計時我們必須為每一個變數、常數、方法、函式、程序、類別及物件等命名，以上所有變數、常數、方法等名稱，統稱為程式語言的識別字。VB 的識別字命名規則如下。

1. 識別字必須是以字母（大小寫的 A 至 Z）或底線（_）開頭。例如：以下是一些合法的識別字。

```
a
I
sum
_sum
InCome
```

以下是一些非法的識別字。

```
7eleven          ' 不能由數字開頭
+a               ' 不能由加號開頭
$sum             ' 不能由錢字號開頭
```

2. 識別字由字母開頭後，可由字母、數字組合而成，且不得包含空白與符號。例如：以下是一些合法的識別字。

```
a123
a123b
```

以下是一些非法的識別字。

```
A C              ' 不能含空白
c+3              ' 不能含加號
Age#3            ' 不能含井字號
```

3. 識別字如果是以底線開頭，必須含有至少一個英文字母或十進位數字。

4. 識別字長度不可超過 1023 個字元。

5. 識別字不得使用保留字，如 If、For 等。若誤用則電腦會出現「關鍵字作為識別項無效」等錯誤訊息。

6. 識別字要盡量用有意義的單字。例如：StudentNumber 或 AverageIncome。除非有效範圍極短的變數才用 x、i、或 a 等當識別字，也千萬不要用像 k23erp 這種沒意義又難記的識別字。

7. 識別字有多個單字時，中間可以加上底線（_）。例如：上例的 StudentNumber 可寫成 Student_Number，若擔心打字不靈光亦可寫成 Stu_Num、stu_num、StuNum 或 stunum，其中 StuNum 又稱為駝峰表示法，因為大寫字母看起來像駝峰一樣，可以避免鍵入底線的困擾、且提昇閱讀效率。

2-2 資料的表示及資料型別

一、資料的表示

電腦的主要工作就是處理資料，VB 所能處理的資料為整數、浮點數、布林值、字串及時間等，分別說明如下：

⊙ 整數

VB 可以處理的整數有三種進位方式，分別是十進位、八進位（Octal）及十六進位（Hexadecimal）等。十進位的表示方式同我們平常的書寫方式，如 25；八進位的表示方式為在數字的前面加上（&O），例如：&O12 則為十進位的 10；十六進位的表示方式為在數字的前面加上（&H），例如：&H12 則為十進位的 18。

⊙ 浮點數

數字中含有小數點或指數的稱為浮點數或實數。以指數為例，E 或 e 表示 10 的次方，例如 0.0023、2.3E-3 及 2.3e-3 均是表示相同的浮點數；又例如 2.3E+2 則代表 230。

⊙ 布林值

VB 使用 True 與 False 表示布林值。

⊙ 字串

不需加減乘除等運算的資料，通常用字串表示。且需使用雙引號（"）將字串的兩旁括起來。例如：以下是一些合法的字串。

```
"This is a book"
" 張中立 "
```

⊙ 時間

日期與時間均應使用井號（ # ）括住。例如：

```
# 3/12/02 #
```

即表示 2002 年 3 月 12 日。又例如：

```
# 4/18/98 3：21：43 PM #
```

表示 1998 年 4 月 18 日 15 時 21 分 43 秒。

二、資料型別

VB 所能處理的資料如上所述，這些資料有的很佔空間，有的很小，接下來要討論的是電腦應如何儲存以上資料，才能正確的儲存資料，並達到快速搬運的效果。就如同冰箱要用大紙箱裝，小滑鼠用小紙盒裝，才能節省空間，並快速的搬運。當然，假如每個滑鼠都使用冰箱的紙箱包裝，當然也可達成任務，但是卻非常浪費空間，搬運也非常費時。而 VB 為了有效地使用記憶體儲存資料，並儘量節省搬運時間，於是規劃了以下的資料型別。

線上查詢請輸入 Data Types，再點選 Data Type Summary，洋洋灑灑共得到 17 種資料型別（或型態），常令初學者眼花繚亂，筆者將常用的 5 種摘錄如下，讀者若發現範圍不足時，再自行尋找適當資料型別。

	資料型別（或型態）	簡稱	儲存大小	範圍
1	Boolean	布林	2 個位元組（bytes）	True 或 False
2	Integer (or) Short	數值	2 個位元組	-32768 到 32767
3	Single	單精度浮點數	4 個位元組	(+/-)3.4 e (+/-)38
4	String	字串	依字串長度	1 到大約 65,400
6	Date	時間與日期	8 個位元組	西元 100 年 1 月 1 日至 西元 9999 年 12 月 31 日

2-3 變數與常數的宣告

一、變數的宣告

變數的功能是用來輸入、處理及儲存外界的資料，而變數在使用以前均要於保留字 Dim 之後宣告，才可使用。在一些舊式的 Basic 語言中，變數並不需要事先宣告，卻也帶來極大的困擾。以下式子即為變數未宣告的後果，編譯器便無法回應使用者在拼字上的錯誤，而造成除錯上的困難。

```
student = studend+1
```

以上敘述若事先宣告 student 如下：

```
Dim student As Short
```

則編譯器遇到 studend 時，便會提醒使用者此 studend 並未宣告的錯誤訊息，提醒使用者補宣告或注意拼字錯誤。其次，變數宣告的優點是可配置恰當的記憶體而提高資料的處理效率。例如：有些變數的值域僅為整數，則不用宣告為浮點數或 Decimal。甚至有些變數的值域小到使用 Byte 型別即可儲存，此時當然宣告為 Byte 型別即可。這也就是讓小東西用小箱子裝、大東西用大箱子裝的道理，如此才能更有效率地管理記憶體，而且提高其搬運速度。

VB 變數的宣告的簡要語法如下：

```
Dim 變數名稱 As 資料型別 [= 初值 ]
```

例如：以下敘述可宣告變數 a 的型別為 Short。

```
Dim a As Short
```

以下敘述可於宣告變數的同時亦可給予初值。

```
Dim a as Short =34
```

若有兩個以上的變數有相同的型別，亦可同時給予宣告，但是變數之間需以逗號（,）隔開。例如：以下敘述可同時宣告變數 b、c 的型別為 String。

```
Dim b,c As String
```

二、常數符號的宣告

跟變數一樣，常數符號（以下簡稱常數）亦需要配置記憶體，與變數不同的是，常數符號正如其名稱所示，在整個程式中都不會改變其值，故稱為常數。程式設計中，有兩種表示常數的方式：(1) 一種是文字式（Literal）。例如：直接以 15 或 3.14159 表示某一常數；(2) 另一種則是本單元所要介紹的常數符號式（Symbolic）。之所以需要常數符號，是因為有些數字在程式中會不斷的重複出現，為了增加程式的可讀性及減少程式鍵入的麻煩，此時即可用一個有意義的符號代替，但必須在符號之前加上保留字 Const，則該符號的值將永遠保持在所宣告的符號中，程式中任何位置均不能改變其值，此稱為常數符號，簡稱常數。例如：於利息的計算，應將利率 RATE 設為常數。

```
Const RATE=0.2
```

則每次要使用利率時，只要填入 RATE 即可（註 常數符號通常將每一個字母均用大寫表示）。待有朝一日必須調整利率時，只要在程式的最前面修改此常數符號的值即可。若未使用常數符號統一此值，則因此常數散落程式各地而無法確保此值的一致性。以下敘述是常數宣告的語法，且不用指派其型別。VB 將會依照所輸入的值，選擇最適當的記憶體配置。

```
Const 常數識別字 = 常數
```

或者，你也可以指派其型別。

```
Const 常數識別字 As 資料型別 = 常數
```

以下敘述即是宣告常數 I = 3。

```
Const I = 3                    ' 不指派 I 的型別
```

或是

```
Const I As Short = 3           ' 指派 I 為整數型別
```

三、變數與常數的有效範圍

在一個大型專案裏，一個專案是由數個表單組合而成。這些表單通常由許多人合力完成，為了避免彼此變數互相干擾，所以必須制訂變數的有效範圍（又稱為生命週期）。例如：張三設 stunum=15，李四又設 stunum=20，則必然造成無法預期的錯誤，茲將變數的有效範圍敘述如下：

任一變數的宣告，若無特殊聲明，均屬於區域變數，其有效範圍僅止於該變數所在的程式區塊。例如：宣告在某個敘述區塊中，如 if 或 for 敘述區塊，則它的有效範圍僅止於該敘述區塊，別的敘述區塊是無法取得該區域變數的值；若宣告於函式中，則它的有效範圍僅止於該函式，其它的函式均無法取得該區域變數的值；若宣告在表單，則其有效範圍僅止於該表單，同樣的道理，其它的表單是無法取得該區域變數的值。例如：以下敘述的 a 變數，它的有效範圍是整個表單，所以 cmd1_Click() 及 cmd2_Click() 均可存取 a 變數。其次，cmd1_Click() 及 cmd2_Click() 函式內宣告 b 與 c 變數，則它們的有效範圍僅止於 cmd1_Click() 及 cmd2_Click() 函式內，只要離開此函式即無法存取這些變數。

最後，雖然 cmd1_Click() 與 cmd2_Click() 函式內各有 d 變數，但此 d 卻各自佔用不同的記憶體，彼此不相互干擾，此即為區域變數的優點。就如同在不同的家庭裡，難免會取用相同的人名，但是到戶政事務所登記時，一定可以分配到不同的身份證號碼。其次，相同的變數名稱，不能出現在同一函式，就如同，同一家庭不會有兩個人同名。

```
Public Class Form1
    Dim a As Short
    Private Sub Button1_Click(…) Handles Button1.Click
        Dim b, d As Short
    End Sub

    Private Sub Button2_Click(…) Handles Button2.Click
        Dim c, d As Short
    End Sub
End Class
```

動手做

識別字更正。請開啟主控台應用程式，並鍵入以下程式，若有錯誤，並完成更正。

```
Module Module1
    Sub Main()
        Dim a As Short
        Dim 7a as Short
        Dim +a as Short
        Dim $sum as byte
        Dim _a As Boolean
        Dim a c as Char
        Dim a+c as string
        Dim if as Short
    End Sub
End Module
```

2-4 運算子

可以對運算元（Operand）執行特定功能的特殊符號稱為運算子。一般而言，運算子可分為以下幾類：指派（Assignment）運算子、算術（Arithmetic）運算子、關係（Relational）運算子、邏輯（Logical）運算子、字串（String）運算子、複合指派運算子及位元操作（Bitwise）運算子。

而每一種運算子都可以再細分為一元（Unary）運算子與二元（Binary）運算子。一元運算子只需要一個運算元就可以操作，而二元運算子則需要兩個運算元才能夠操作。

在以下單元中，我們除了檢視各種不同的運算子功能外，還將討論運算子的優先順序（Precedence）與結合律（Associativity）。優先順序是用來決定同一式子擁有多個運算子時，每一個運算子進行運算的順序；而結合律則決定了在同一敘述中，相同優先順序的運算子執行順序。

一、指派運算子（Assignment operator）

指派（或稱指定）運算子的符號為 =（只有 Pascal 與 Delphi 採用：=），其作用為將運算符號右邊運算式的值指派給運算符號左邊的運算元。所以，以下敘述 sum = a + b，是將 a + b 的值指派給 sum。

```
sum = 0 : a = 3 : b = 5          ' 同一行內有多個敘述時，需以冒號連接
sum = a + b
```

上式與數學的等號是不同的，所以不要一直懷疑為什麼 0 會等於 8。其次，你不能將常數放在指派運算子的左邊。例如：

```
8 = x
```

為一個不合法的敘述，但以下敘述將常數 8 指派給變數 x 為合法的。

```
x = 8
```

以下敘述

```
a=a+1
```

則是將變數 a 之值取出，加一，再指派回 a。所以 a 之值是 4。

二、算術運算子（Arithmetic operators）

算術運算子用來執行一般的數學運算，包括取正數（+）、取負數（–）、加（+）、減（–）、乘（*）、除（/）、整數除法（\）、次方（^）及取餘數（Mod）等，如右表所示。

運算子符號	功能	範例	結果
+	取正數	+2	
–	取負數	–3	
+	加	6 + 2	8
–	減	6 – 2	4
*	乘	6 * 2	12
/	除	7 / 2	3.5
\	整數除法	7 \ 2	3
^	次方	3^2	9
Mod	取餘數	6 Mod 3	0

動手做

請於命令視窗（點選功能表的「檢視／其他視窗／命令視窗」）完成以上運算。例如：

```
? 7 mod 3
```

結果如右圖。問號（?）代表輸出結果，在命令視窗請直接用問號；在主控台應用程式請用 Console.Write(); 在 Windows Forms 請用 Label、TextBox 等控制項的 Text 屬性輸出結果。

三、關係運算子（Relational operators）

關係運算子又稱為比較運算子，用於資料之間的大小比較，比較的結果可得到邏輯的 True 或 False，右表是 VB 中的關係運算子符號。

運算子符號	功能	範例	結果
=	等於	2 = 3	False
<>	不等於	2 <> 3	True
<	小於	2 < 3	True
>	大於	2 > 3	False
<=	小於或等於	2 <= 3	True
>=	大於或等於	2 >= 3	False

動手做

請於命令視窗完成以上運算，如右圖。

四、邏輯運算子（Logical operators）

當同一個運算式同時具有兩個以上的關係運算子時，則每兩個關係運算子之間必須使用邏輯運算子連結。VB 的邏輯運算子如右表所示。例如：若要判斷 x 是否介於 1 和 7 之間：

運算子符號	功能	範例	結果
And	And	(2 > 3) And (3 = 1)	False
Or	Or	(2 > 3) Or (3 <> 1)	True
Xor	互斥或	True Xor True	False
		False Xor False	False
		True Xor False	True
Not	Not	Not True	False
		Not False	True

```
1 < x < 7
```

則其 VB 敘述如下：

```
If (x > 1) And (x < 7)
```

動手做

請於命令視窗完成以上運算。

五、字串運算子（String operator）

前面關係運算子的 =、<>、>、<、>= 及 <= 均適用於字串的大小比較，而右表的字串連結運算子（&、+），則可連結兩個字串。

運算子符號	功能	範例	結果
&	字串連結	"abc" & "xyz"	abcxyz
+	字串連結	"123" + "456"	123456

動手做

請於命令視窗完成以上運算。

六、位元運算子

將指派的運算元先轉為二進位，再逐一執行位元之間的運算，稱為位元運算子。VB 常用的位元運算子說明如下：

⊙ And 運算子

兩個運算元皆為 1，才能得到 1，否則為 0。例如：

```
a = 6              ' 二進位為 (00000110)
b = 255            ' 二進位為 (11111111)
c = a And b
? c
```

結果是 6，其二進位是（00000110）。問號（?）代表輸出結果，在命令視窗請直接用問號；在主控台應用程式請用 Console.Write(); 在 Windows Forms 請用 Label 或 TextBox 控制項的 Text 屬性輸出結果。

⊙ Or 運算子

兩個運算元只要有一個為 1，就得到 1，否則為 0。例如：

```
a = 6              ' 二進位為 (00000110)
b = 255            ' 二進位為 (11111111)
c = a Or b
? c
```

結果是 255，其二進位是（11111111）。

⊙ Xor 運算子

兩個運算元不同時，就得到 1，否則為 0。例如：

```
a = 6              ' 二進位為 (00000110)
b = 255            ' 二進位為 (11111111)
c = a Xor b
? c
```

結果是 249，其二進位是（11111001）。

⊙ Not 運算子

此為單一運算子，將運算元的 1 轉為 0，0 轉為 1。例如：

```
a = 6                ' 二進位為 (00000110)
a = Not a
? a
```

結果是 –7，其二進位是（11111001）。

動手做

請於主控台應用程式完成以上運算。問號（?）代表輸出結果，在主控台應用程式請用 Console.Write(); 在 Windows Forms 請用 Label 或 TextBox 控制項的 Text 屬性輸出結果。

⊙ 雜項運算子

以下是一些無法歸類的運算子，列表如右。這些運算子將會在往後各章節陸續介紹。

運算子符號	使用範例	說明
.	x.y	連結物件與類別成員
( )	a( )	一維陣列
(,)	a(,)	二維陣列
( )	AA("String")	函式 AA 的叫用

⊙ 運算子的優先順序（Precedence）

於較複雜的運算式中，通常同時存在許多運算子，此時就需要定義其優先順序，右表即是 VB 關於運算子的優先順序表。

例如：

```
x = x + y * z
```

則等效於以下敘述。

```
x = x + (y * z)
```

分類	運算子	優先等級	結合律
指數	^	1	
一元（Unary）	+, -（取正負值）	2	
乘法，除法	*, /	3	左結合
整數除法	\	4	左結合
取餘數	Mod	5	左結合
加法，減法	+, - 字串連結（+）	6	左結合
字串連結	&	7	左結合
關係	=, <>, <, >, <=, >=,	8	左結合
邏輯，位元	Not	9	左結合
邏輯，位元	And	10	左結合
邏輯，位元	Or	11	左結合
邏輯，位元	Xor	12	左結合
指派與複合指派	=, +=, -=, *=, /=, &=	13（最低）	右結合

又例如：

```
z = x > 2 And y > 3
```

則等效於以下敘述。

```
z = (x > 2) And (y > 3)
```

動手做

令 x＝1、y＝2、z＝3，請於主控台應用程式完成以上運算。

七、運算子的結合律（Associativity）

當同一敘述包含相同優先順序的運算子時，就需要定義結合律。結合律有右結合與左結合。於上表的運算子優先順序中，我們已標註運算子的結合律。例如：

```
x - y + z
```

則等效於

```
((x - y) + z)
```

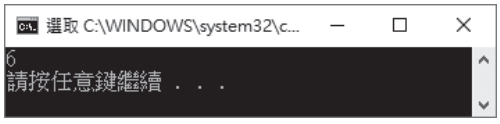
範例 2-4a

請將以下數學敘述以 VB 敘述表示，並在主控台應用程式執行結果。

三角形三邊長分別是 a＝3、b＝4、c＝5，先令 $d=\frac{a+b+c}{2}$，

則其面積 $s=\sqrt{d(d-a)(d-b)(d-c)}$。

執行結果



程式列印

```
Module Module1
    Sub Main()
        Dim a, b, c As Short
        a = 3
        b = 4
        c = 5
        Dim d, s As Single
        d = (a + b + c) / 2
        s = (d * (d - a) * (d - b) * (d - c)) ^ (1 / 2)
        Console.WriteLine(s)
    End Sub
End Module
```

動手做

1. 平面座標兩點的距離 d。

   令 $x1=3$、$y1=0$、$x2=0$、$y2=4$，則兩點距離 $d=\sqrt{(x1-x2)^2+(y1-y2)^2}$，

   請於主控台應用程式執行結果。

2. 一元二次方程式 $ax^2+bx+c=0$ 的解。

   令 $a=2$、$b=-5$、$c=2$，則 $x1=\dfrac{-b+\sqrt{b^2-4ac}}{2a}$、$x2=\dfrac{-b-\sqrt{b^2-4ac}}{2a}$，

   請於主控台應用程式執行結果。

2-5 敘述與運算式

一、敘述（Statement）

凡是控制執行的順序、對運算式取值或不作任何事，均可稱為敘述或陳述式。以下式子即是一個敘述。

```
    sum = sum + 1
```

⊙ 敘述區塊（Block Statement）或複合敘述（Compund Statement）

在任何可以放單一敘述的地方，也就能放敘述區塊，敘述區塊亦稱為複合敘述。一個複合敘述是由兩個以上的敘述組合而成，如下所示。

```
t = a
a = b
b = t
```

二、運算式（Expression）

任何可求得值的式子，均稱為運算式。例如：5+3 會傳回一個數值，所以 5+3 是一個運算式。一般而言，可以放在等號右邊的東西，都可以稱為運算式。例如：以下括號內的東西均稱為運算式。

```
i = 0 : a = 3 : b = 5
Console.WriteLine (i+1)        '1
Console.WriteLine (a+b)        '8
If (a>b)          'False
While (a=b)       'True
```

三、註解（Comments）

適當的程式註解能增加程式的可讀性。其次，註解是給人看的，編譯器均不予理會註解的內容。VB 的註解有兩種表示方式，凡是單引號（'）或 Rem 以後的敘述，VB 均視為註解。例如：以下敘述的「將 y 值累加至 sum」為註解，不會被執行。

```
sum = sum + y                  ' 將 y 值累加至 sum
Rem 將 y 值累加至 sum
```

其次，VB 編譯器會將註解轉為綠色，如下圖所示。