

# 呼叫功能

第一種寫法：

建立一個"變數"，變數內建立運算式(最後需要回傳值)

## 回傳(return)

在變數和函式中，基本上每一個有進行運算的資料，就需要回傳目標，如果不需回傳的無類型(void)類型，則不會進行回傳。void在英文詞彙中代表虛無的意思，程式中則代表運算完後不會回傳，計算本身就是目的。

回傳指在一切計算完成後，完成計算的值會給誰，有點類似我丟了檸檬進去榨汁機，而我需要檸檬汁，所以我必須要指定回傳給我的是檸檬汁，無論是函式或變數，在計算中可能會有非常多的數值參雜在內，回傳的對象除了自己本身以外，還可以是其他值。

簡單看過一次最好理解，以下是範例：

```
//普通的宣告
private int a;
//同樣意思，使用return來呈現
private int a
{
    return;
}
```

```
using System;          //引用系統類別
using System.Linq.Expressions;
using System.Text;

// 建立一個class multiplication(乘法)，裝入 "set_an_num"(給予一個數值)
public class multiplication
{
    // 建立一個欄位 nummm，指定型態是int
    public int nummm;
```

```

// 建立一個欄位 set_an_num, 指定型態是int, 並且會接收一個值(num_input, 指定型態是int)
public int set_an_num(int num_input)
{
    // 輸入的數值(num_input) 乘以2, 並且賦值給 nummm
    nummm = num_input * 2;

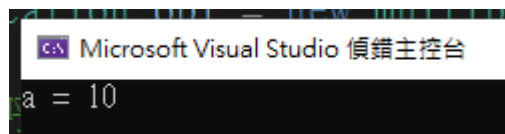
    // 運算完之後, 回傳 nummm
    return nummm;
}

// 主程式
static void Main()
{
    // 實體化 上面的 multiplication, 並且 命名為 obj
    multiplication obj = new multiplication();

    // 用obj呼叫內部變數 set_an_num, 並且給予該變數一個值(5)
    var a = obj.set_an_num(5);

    Console.WriteLine("a = " + a);
}
}

```



=====

第2種寫法：

建立一個”函式”，內部放入運算式(最後不需要回傳值)

屬於”無類型(void)回傳”

## 無類型(void)回傳

這個也很好懂，就是這個值不需要回傳，通常用於函數，也就是計算本身就是目的，在資料型態的宣告前面添加「void」即可。

```
private void method01()
{
    //計算的內容
}
private void method02(int a,int b)
{
    //計算的內容(必定包含a&b)
}
```

```
using System;          //引用系統類別
using System.Linq.Expressions;
using System.Text;

// 建立一個class multiplication(乘法)
public class multiplication
{
    // 建立一個欄位 num_2, 指定型態是int
    public int num_2;

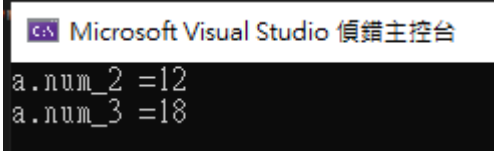
    // 建立一個欄位 num_3, 指定型態是int
    public int num_3;

    // 定義 函式 會接收一個數值((num_input, 指定型態是int)
    public multiplication(int num_input)
    {
        // 接收到的數值 乘以2, 並且賦值給 nummm, "這邊不需要回傳return", "無類型回傳"
        num_2 = num_input * 2 ;
        num_3 = num_input * 3 ;
    }

    static void Main()
    {
        // 呼叫功能, 並給予數值(6), 定義成a, a可以呼叫該功能內的變數
        var a = new multiplication(6);

        // 呼叫功能內的變數num_2
        Console.WriteLine("a.num_2 =" + a.num_2);
    }
}
```

```
// 呼叫功能內的變數num_3  
Console.WriteLine("a.num_3 =" + a.num_3);  
}  
}
```



Microsoft Visual Studio 偵錯主控台

```
a.num_2 =12  
a.num_3 =18
```