

開始撰寫 Visual C# 程式

- 1-1 認識 Visual C# ○
- 1-2 安裝 Visual Studio Community ○
- 1-3 建立 Windows Forms 應用程式 ○
- 1-4 Visual C# 程式碼撰寫慣例 ○
- 1-5 使用 MessageBox.Show() 方法 ○
- 1-6 建立主控台應用程式 ○
- 1-7 使用主控台輸入 / 輸出 ○

1-1 認識 Visual C#

C# (唸做 C sharp) 是 Microsoft 公司根據 C/C++ 所發展出來的程式語言，具有簡潔、型別安全、物件導向等特色，可以用來快速開發應用程式。C# 的語法類似 C/C++ 和 Java，因此，熟悉 C/C++ 或 Java 的人很快就能學會 C#。

C# 修改了 C/C++ 一些複雜的功能，例如命名空間 (namespace)、類別 (class)、列舉 (enumeration)、重載 (overloading)、結構化例外處理等，同時刪除了 C/C++ 的某些功能，例如多重繼承 (multiple inheritance)、巨集 (macro)、虛擬基底類別 (virtual base class) 等，但也提供了 C/C++ 所沒有的功能，例如可為 null 的型別 (nullable type)、委派 (delegate)、匿名方法、泛型 (generic)、部分類別 (partial class)、Iterator、匿名型別、擴充方法、隱含型別等。

身為一個物件導向程式語言，C# 支援封裝 (encapsulation)、繼承 (inheritance)、多型 (polymorphism)、介面 (interface)、覆蓋 (override)、重載 (overload)、虛擬函式 (virtual function)、運算子重載等功能。

Visual C# 是 Microsoft 公司的 C# 語言實作，同時 Microsoft 公司亦針對 Visual C# 推出一個功能強大的整合開發環境 **Visual Studio 2017**，包括互動式開發環境、視覺化設計工具、程式碼編輯器、編譯器、專案範本、偵錯工具等。Visual C# 完全整合 .NET Framework 和 CLR，能夠快速建立 Windows Forms 應用程式、ASP.NET Web 應用程式、native Android App、native iOS App、Azure 雲端服務等。

註¹：**.NET Framework** 是針對 Windows、Windows 市集、Windows Phone、Windows Server 和 Microsoft Azure 建立應用程式的開發平台，包括 Visual Basic、C#、C++ 等程式語言、CLR (Common Language Runtime)，以及廣泛的類別庫。

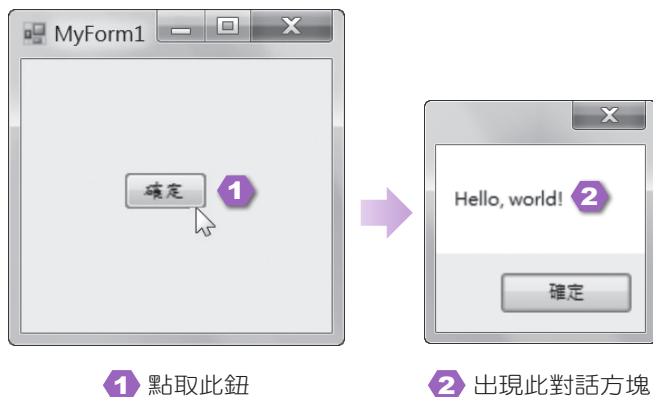
註²：**CLR** (Common Language Runtime，共通語言執行環境) 除了負責執行程式，還要提供記憶體管理、執行緒管理、安全管理、版本管理、例外處理、共通型別系統 (CTS，Common Type System) 與生命週期監督等核心服務。

1-3 建立 Windows Forms 應用程式

Visual C# 是一個視覺化的程式開發工具，其設計流程與傳統的程式語言並不完全相同，但可以簡單歸納成下列幾個步驟：

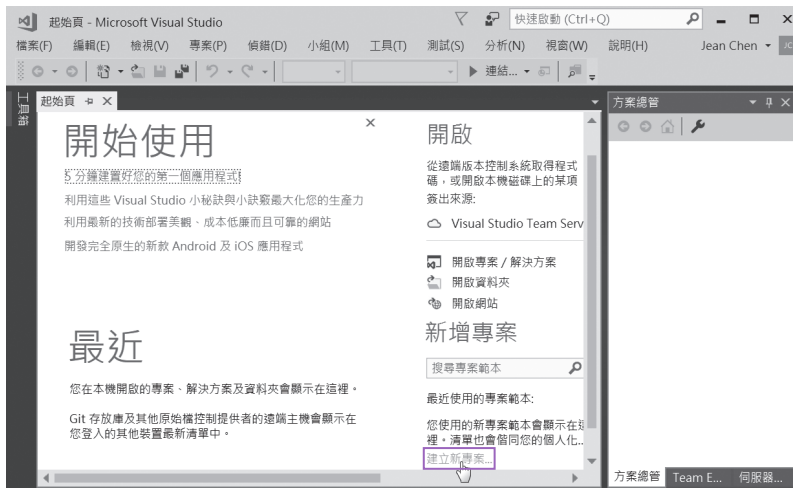
1. **建立專案**：在 Visual Studio 選取 [檔案] \ [新增] \ [專案]，以建立專案，任何 Visual C# 程式都必須放在專案內。
2. **建立使用者介面**：從 [工具箱] 選擇控制項加入表單，以建立使用者介面。舉例來說，假設使用者介面有一個按鈕，那麼可以在表單上放置一個 Button 控制項。
3. **自訂外觀**：透過 [屬性視窗] 設定表單與控制項的外觀，例如表單的大小、標題列的文字、按鈕的大小、文字、字型等屬性。
4. **加入 Visual C# 程式碼**：針對可能產生事件的控制項撰寫處理程序。
5. **建置與執行程式**：按 [F5] 鍵進行建置與執行。

為了讓您瞭解 Visual C# 程式的設計流程，我們先做個簡單的例子，之後再講解 Visual C# 的語法。在這個例子中，程式一開始會顯示如左下圖的視窗，使用者只要點取 [確定] 按鈕，就會出現另一個對話方塊，上面顯示著 "Hello, world!"。若要結束程式，關閉這兩個視窗即可。

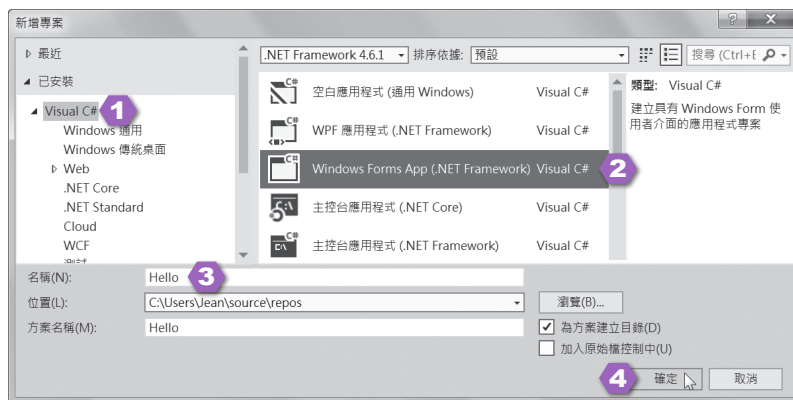


1-3-1 新增專案

1. 按 [開始] \ [Visual Studio 2017]，啟動 Visual Studio，然後在起始頁點取 [建立新專案] 或選取 [檔案] \ [新增] \ [專案]。



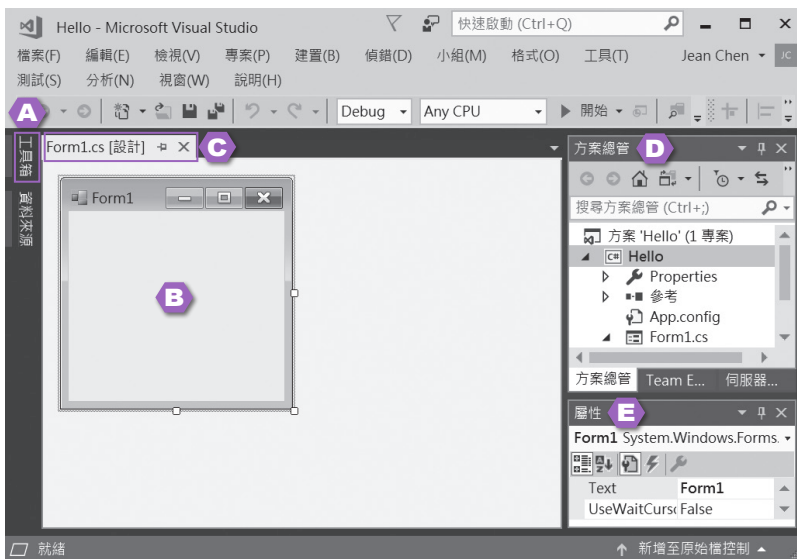
2. 依照下圖操作，新增一個名為 Hello 的專案。



- 1 選擇 [Visual C#]
- 2 選擇 [Windows Forms App]
- 3 輸入專案名稱，例如 Hello
- 4 按 [確定]

3. Visual Studio 會根據步驟 2 輸入的專案名稱 **Hello**，建立副檔名為 **.csproj** 的專案檔及副檔名為 **.sln** 的方案檔，而且預設的存檔路徑為 **C:\Users\使用者名稱\source\repos\Hello**。您可以將**專案 (project)** 視為建置後的一個可執行單位，而大型應用程式往往是由多個可執行單位所組成，因此，Visual Studio 是以一個**方案 (solution)** 管理一個或多個專案。

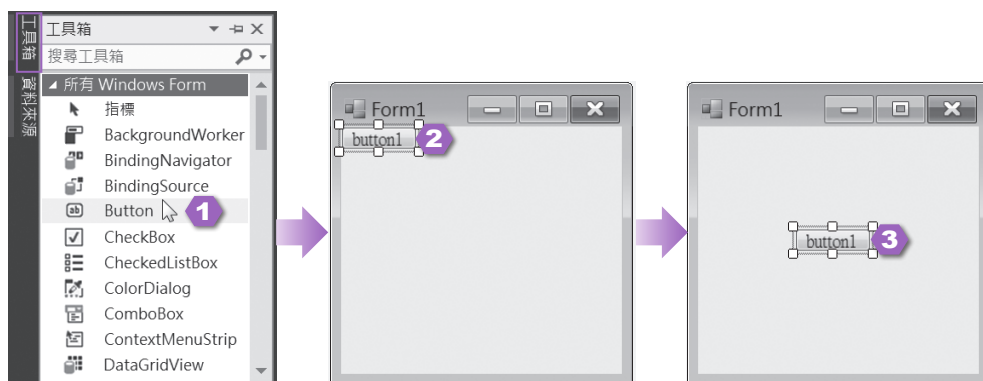
在新增專案後，Visual Studio 的畫面中間有一個名稱為 **Form1** 的表單，這就是 **Windows Forms 設計工具**，用來設計應用程式的介面。若沒有看到設計工具，可以在 **[方案總管]** 內找到 **Form1.cs**，然後按兩下。



- A** 點取此標籤可以顯示工具箱
- B** Windows Forms 設計工具 (若要調整表單的大小，可以拖曳表單四周的空心小方塊)
- C** 此處的標籤用來切換表單或關閉表單
- D** 方案總管用來管理方案內的專案或檔案 (若沒有看到方案總管，可以選取 **[檢視] \ [方案總管]**)
- E** 屬性視窗用來設定表單或按鈕、圖片、標籤等控制項的屬性 (若沒有看到屬性視窗，可以選取 **[檢視] \ [屬性視窗]**)

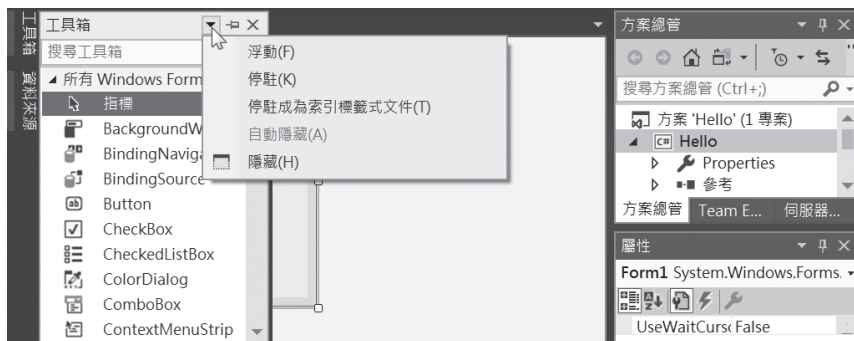
1-3-2 建立使用者介面（在表單上放置控制項）

在這個例子中，我們將利用工具箱的 Button 控制項在表單上放置按鈕，請依照下圖操作。



- ❶ 點取 [工具箱] 標籤，然後找到 Button 控制項並按兩下
- ❷ 出現一個按鈕，上面預設的文字是按鈕名稱
- ❸ 將按鈕拖曳至適當的位置，若要調整大小，可以拖曳四周的空心小方塊，若要刪除，可以按 [Del] 鍵

工具箱預設會自動隱藏到視窗左側，只留下一個標籤，若要固定顯示工具箱，可以點取橫向的大頭針圖示，令它變成直立的，或點取向下箭頭，然後選擇讓視窗浮動在視窗內、停駐在視窗左側、以標籤頁顯示或自動隱藏。



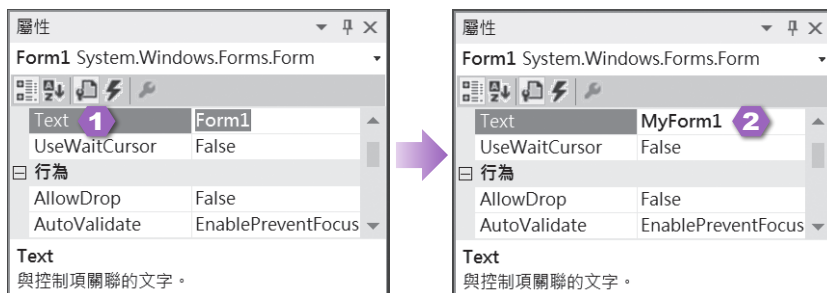
1-3-3 自訂外觀 (設定表單與控制項的屬性)

接著，我們要根據下表設定表單與按鈕的屬性。

物件	屬性	值	說明
表單	Text	我的第一個程式	表單的標題列文字
按鈕	Text	確定	按鈕的文字
	Font	標楷體、9 點、標準	按鈕的文字字型

設定表單的屬性

1. 選取表單，然後移動屬性視窗的捲軸，找到 [Text] 屬性，在 [Text] 屬性的名稱按兩下，此時，[Text] 屬性的值會呈現藍色反白。
2. 輸入新的屬性值 "MyForm1"，表單的標題列文字會由原來的 "Form1" 變成 "MyForm1"。若輸入至一半想取消，可以按 [Esc] 鍵。



① 在 Text 屬性按兩下

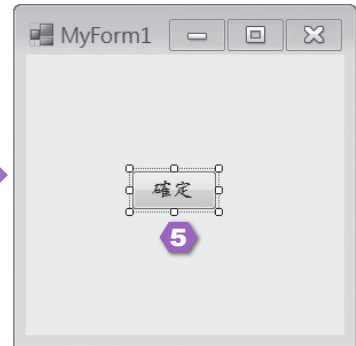
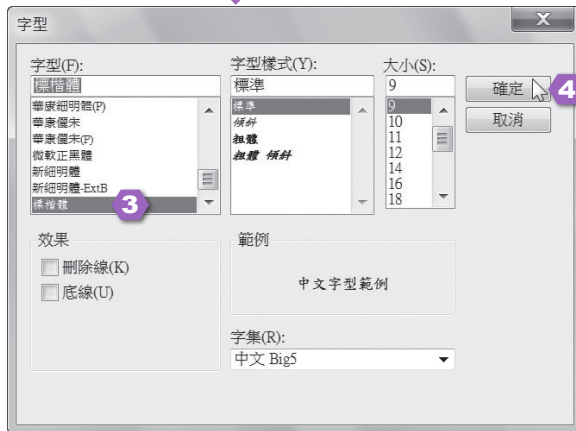
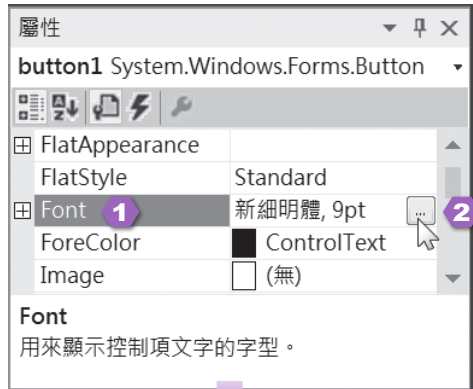
② 輸入新的屬性值

設定按鈕的屬性

1. 選取按鈕，然後在屬性視窗內將 [Text] 屬性的值設定為 "確定"，按鈕上的文字會由原來的 "button1" 變成 "確定"。



2. 選取按鈕，然後移動屬性視窗的捲軸，找到 [Font] 屬性，在 [Font] 屬性的名稱按一下，再點取 [...] 按鈕，螢幕上會出現 [字型] 對話方塊，請從中選擇字型為 [標楷體]、字型樣式為 [標準]、大小為 [9]，最後按 [確定]，按鈕上的文字會由原來的新細明體變成 9 點大小、標準樣式的標楷體。

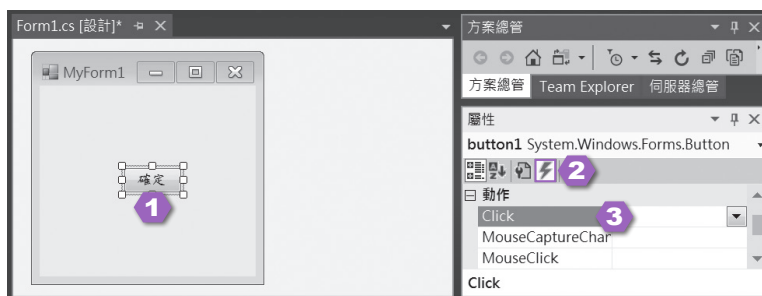


- 1 在 Font 屬性按一下
- 2 點取此鈕
- 3 選取 [標楷體]
- 4 按 [確定]
- 5 設定結果

1-3-4 加入 Visual C# 程式碼

現在，我們要針對這個例子的 " 確定 " 按鈕撰寫事件程序，讓使用者一點取 " 確定 " 按鈕，就出現另一個對話方塊，上面顯示著 "Hello, world!"。

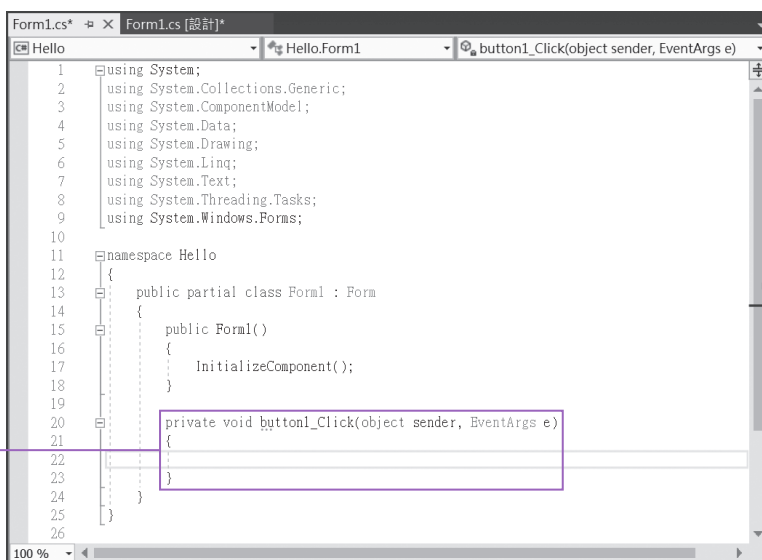
1. 首先，選取欲撰寫事件程序的控制項，例如 " 確定 " 按鈕；接著，點取屬性視窗的 [事件] 按鈕，然後在欲處理的事件按兩下，例如 [Click]。



- 1 選取控制項 2 點取 [事件] 按鈕 3 在 Click 事件按兩下

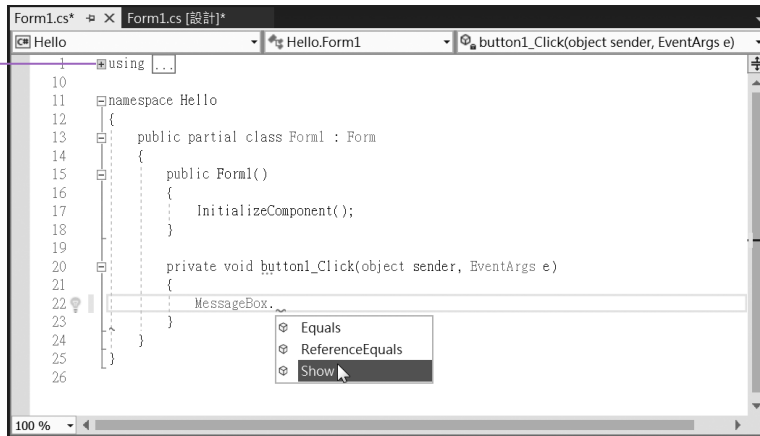
2. Visual C# 自動產生下列程式碼，當使用者點取 button1 按鈕時，系統會產生一個 Click 事件，進而呼叫 button1_Click() 方法做處理。

這個方法為欲撰寫的事件程序



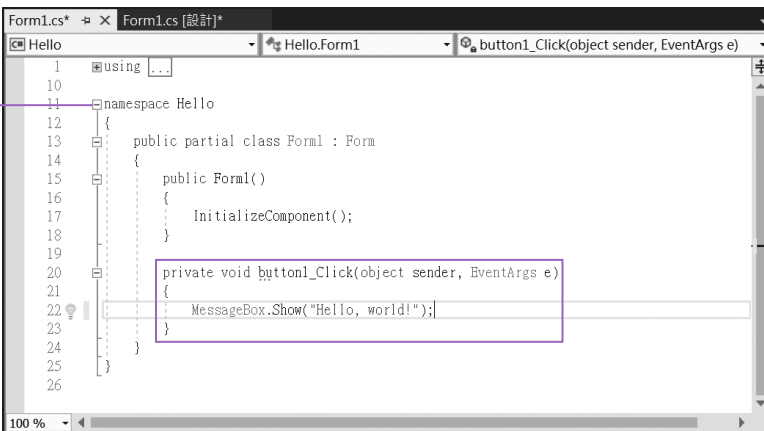
- 將插入點移到 `button1_Click()` 方法裡面，然後輸入程式碼，注意 C# 會區分英文字母的大小寫。在輸入到 `MessageBox` 時，螢幕上會自動出現一個清單，裡面列出 `MessageBox` 類別的方法，此為 **IntelliSense** 功能，目的是讓程式設計人員不用牢記一堆方法或屬性。您可以輸入 `Show`，也可以從清單中找到 `Show` 方法，然後按兩下，`Show` 就會出現在程式碼。

為了方便閱讀，我們將 using 區塊折疊起來，裡面有應用程式所匯入的命名空間。



- 繼續輸入程式碼，直到將 `MessageBox.Show("Hello, world!");` 輸入完畢，記得在右括號的後面輸入分號，做為此敘述的結尾。

點選減號會摺疊程式碼；點選加號會展開程式碼。

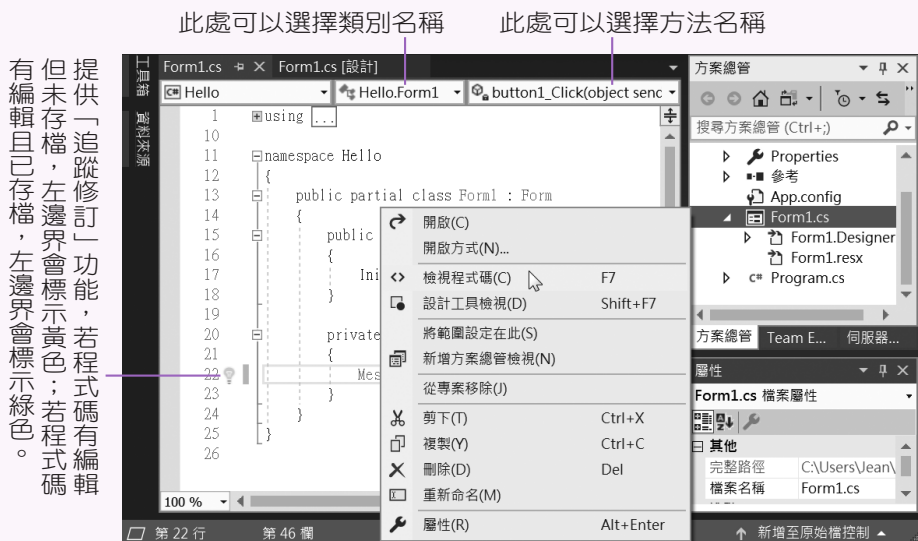




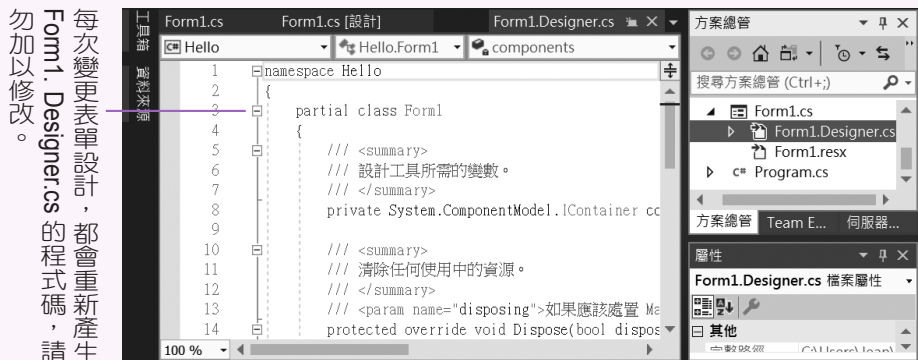
備註

關於程式碼視窗

只要在方案總管內找到要檢視程式碼的檔案，然後按一下滑鼠右鍵，選取 [程式碼檢視]，就能開啟程式碼視窗。

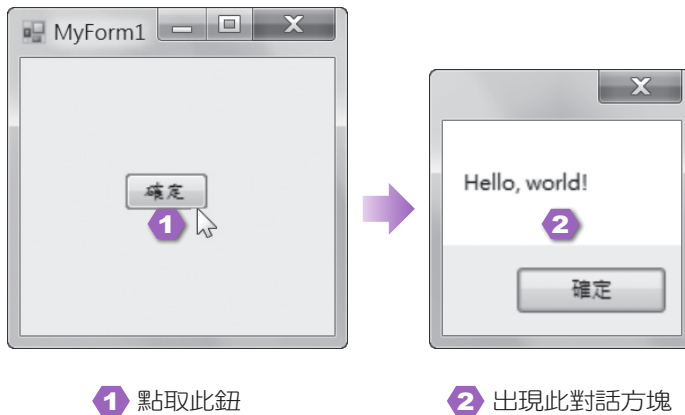


此外，由於 Visual C# 提供**部分類別** (partial class) 功能，因此，一些自動產生的程式碼都被放進 Form1.Designer.cs，而不會出現在 Form1.cs。



1-3-5 建置與執行程式

我們的第一個 Visual C# 程式寫好了，趕快來執行看看吧！請按 [F5] 鍵或點取標準工具列的 [開始] 按鈕，Visual Studio 會先進行建置，確定沒有錯誤，就會出現如下的執行結果，而建置完畢的可執行檔則是放在該專案資料夾內的 bin 子資料夾，若要結束程式，關閉這兩個視窗即可。





請注意，應用程式在執行之前都必須先經過建置，您可以按 [F5] 鍵或點取標準工具列的 [開始] 按鈕進行建置與執行。若只要進行建置，可以選取 [建置] \ [建置方案]，一旦建置的過程產生錯誤，就會顯示在錯誤清單，例如下圖是我們故意遺漏敘述後面的分號所產生的錯誤清單。

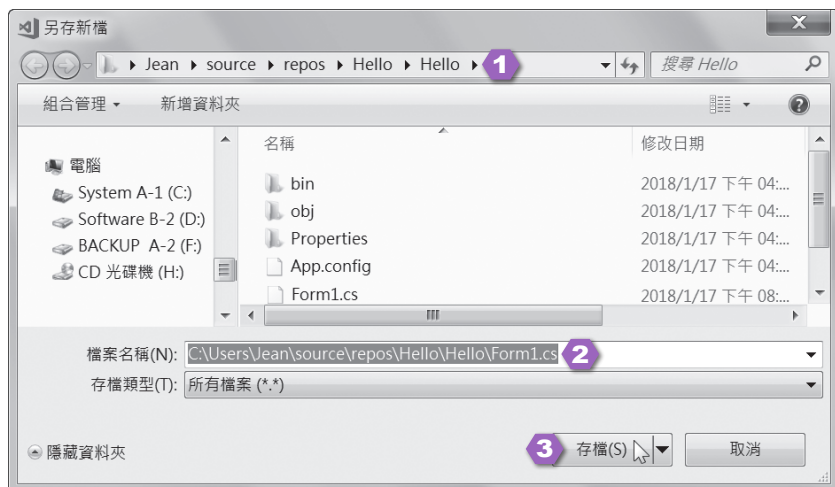


在錯誤按兩下會跳到產生錯誤的程式碼

若沒有看到此視窗，可以選取 [檢視] \ [錯誤清單]

1-3-6 儲存檔案、專案與方案

- ❖ 若要儲存目前正在編輯的檔案，可以點取標準工具列的  [儲存] 按鈕；若要儲存檔案、專案與方案，可以點取標準工具列的  [全部儲存] 按鈕。
- ❖ 若要將正在編輯的檔案以其它名稱儲存，可以選取 [檔案] \ [另存 XXX 為]，XXX 為檔案名稱，然後在 [另存新檔] 對話方塊中進行儲存。




1 選擇儲存路徑

2 輸入新檔名

3 按 [存檔]

1-3-7 關閉檔案、專案與方案

- ❖ 若只要關閉 Windows Forms 設計工具或目前正在編輯的檔案，可以點取 Windows Forms 設計工具或程式碼視窗右上角的  [關閉] 按鈕。
- ❖ 若要關閉專案與方案，可以選取 [檔案] \ [關閉方案]，此時如未存檔，螢幕上會出現對話方塊詢問是否儲存變更，按 [是] 表示存檔再關閉，按 [否] 表示不存檔就關閉，按 [取消] 表示取消關閉的動作。

1-3-8 開啟檔案、專案與方案

- ❖ 若要開啟專案或方案，可以選取 [檔案] \ [開啟] \ [專案 / 方案]，然後在 [開啟專案] 對話方塊中選擇所要開啟的專案或方案。



- 1 選擇儲存路徑 2 選擇專案或方案 3 按 [開啟舊檔]

- ❖ 若要開啟的檔案屬於目前開啟的方案，可以在方案總管內找到這個檔案，然後按兩下；若要開啟的檔案不屬於目前開啟的方案，或目前並沒有開啟任何方案，可以選取 [檔案] \ [開啟檔案]，然後在 [開啟檔案] 對話方塊中選擇所要開啟的檔案。



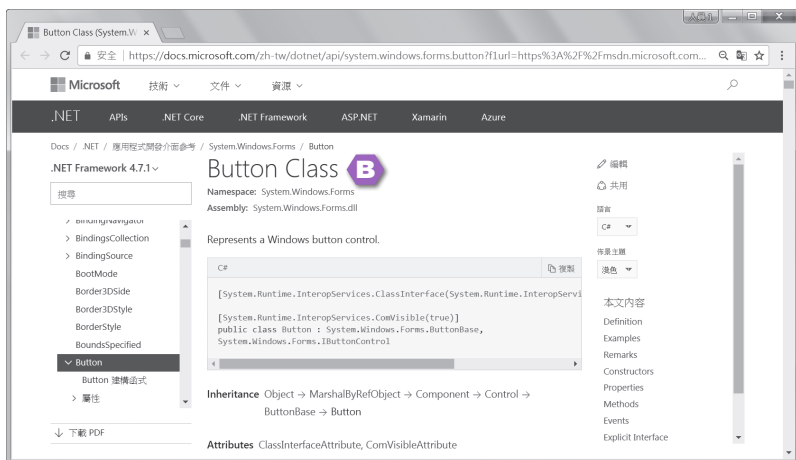
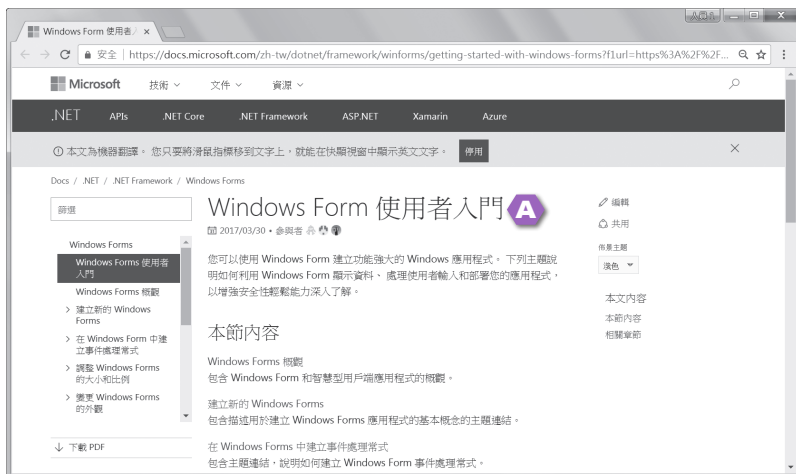
備註

關於 IntelliSense 功能

Visual Studio 的程式碼視窗支援 **IntelliSense** 功能，它會根據您輸入的類別名稱或方法名稱顯示可用的成員清單或參數清單，只要從清單中找到欲使用的成員或參數，然後按兩下，就能插入程式碼。此外，當您輸入方法名稱時，螢幕上會出現語法，而當您輸入錯誤語法時，會出現波浪狀底線，只要將指標移到底線的位置，就會出現說明。若要查看類別或方法的說明，可以將指標移到類別或方法的名稱，然後按 [F1] 鍵，就會開啟相關的說明。

1-3-9 使用線上說明

當您在 Visual Studio 開發 Visual C# 程式時，若對 Visual C# 的語法或控制項有任何疑問，可以選取程式碼或控制項，然後按 **[F1]** 鍵，就會連線到 MSDN 文件庫，讓您查詢相關的線上說明。



A 選取表單後按 **[F1]** 鍵會出現此線上說明

B 選取按鈕控制項後按 **[F1]** 鍵會出現此線上說明

1-4 Visual C# 程式碼撰寫慣例

Visual Studio 是以一個方案 (solution) 管理一個或多個專案 (project)，一個專案又可以包含一個或多個組件 (assembly)，而組件是由一個或多個原始檔 (source file) 編譯而成的 .exe 或 .dll 檔。

至於原始檔是由結構 (structure)、類別 (class) 或介面 (interface) 所組成，而結構、類別或介面是由一行行的敘述 (statement，又稱為陳述式) 所組成，敘述則是由關鍵字 (keyword)、特殊字元 (special character) 或識別字 (identifier) 所組成。

- ❖ **關鍵字**：這是 C# 預先定義的保留字 (reserved word)，包含特殊的意義與用途，程式設計人員必須依照 C# 的規定來使用關鍵字，否則會產生錯誤，例如 `class` 是用來宣告類別的關鍵字，不能用來宣告變數或做其它用途。
- ❖ **特殊字元**：C# 常用的特殊字元不少，例如分號用來標示敘述的結尾、大括號用來標示區塊的開頭與結尾、小括號用來宣告方法或呼叫方法、小數點用來存取類別的成員、中括號用來宣告陣列的大小、`//` 用來標示單行註解、`/* */` 用來標示多行註解。
- ❖ **識別字**：程式設計人員可以自行定義新字做為變數、常數、方法或類別的名稱，例如 `MyClass`、`UserName`、`MouseEventHandler`，這些新字就是屬於識別字。識別字不一定要合乎英文文法，但要合乎 C# 命名規則，我們會在第 1-4-2 節介紹 C# 命名規則。

原則上，敘述是程式內最小的可執行單元，而多個敘述可以組成方法、迴圈、流程控制等較大的可執行單元。

Visual C# 程式碼撰寫慣例涵蓋了程式結構、命名規則、註解、縮排、換行等，雖然不是硬性規定，但遵循這些慣例可以提高可讀性，讓程式更容易偵錯與維護。

1-4-1 Visual C# 程式結構

Visual C# 程式通常會依照如下的順序：

1. **using 指示詞**：這個指示詞用來匯入命名空間或設定命名空間的別名，例如下面的敘述是用來匯入 **System** 命名空間，讓程式可以直接存取 **System** 命名空間所提供的結構、類別或介面：

```
using System;
```

2. **namespace 陳述式**：這個陳述式用來宣告命名空間，前後必須加上大括號標示命名空間的開頭與結尾，裡面可以包含結構、類別、介面或子命名空間等。**.NET** 應用程式的程式碼均包含在命名空間內，而預設的命名空間就是專案的名稱。
3. **class 陳述式**：這個陳述式用來宣告類別，前後必須加上大括號標示類別的開頭與結尾，裡面可以包含欄位、方法、屬性或其它敘述。

Visual C# 程式的敘述區塊不能當作獨立的程式單元，必須放在類別內，下面是一個例子。

```
namespace Hello                // 宣告一個名為 Hello 的命名空間
{
    class Program                // 在 Hello 命名空間內宣告一個名為 Program 的類別
    {
        ...                      // 在 Program 類別內撰寫敘述區塊
    }
}
```

4. **Main() 方法**：這是應用程式的進入點，在應用程式一被執行的當下，就會執行 **Main()** 方法。若應用程式宣告一個以上的 **Main()** 方法，就要在編譯的時候使用 **/main** 編譯器選項指定何者為應用程式的進入點，才不會產生錯誤。

我們可以使用 `Main()` 方法在應用程式一被執行的當下進行初始化的動作，例如宣告變數、建立表單、開啟資料庫連接、判斷哪個表單先載入等。`Main()` 方法有下列幾種形式，您可以視實際情況選擇適合的形式：

- ❖ `static void Main()`：這是最簡單的形式，沒有參數及傳回值。
- ❖ `static void Main(string[] args)`：這種形式接受字串陣列參數，您可以撰寫處理字串陣列參數的敘述。
- ❖ `static int Main()`：這種形式有一個整數型別的傳回值做為程式的結束代碼 (exit code)，例如下面的 `Main()` 方法會執行視窗顯示 "Hello, world!"，然後傳回整數 0 做為結束代碼：

```
static int Main()
{
    System.Console.WriteLine("Hello, world!");           // 顯示 "Hello, world!"
    return 0;                                             // 傳回整數 0 做為結束代碼
}
```

- ❖ `static int Main(string[] args)`：這種形式接受字串陣列參數，而且有一個整數型別的傳回值做為程式的結束代碼 (exit code)。



備註

C# 內建許多關鍵字，例如 `abstract`、`as`、`base`、`bool`、`break`、`byte`、`case`、`catch`、`char`、`checked`、`class`、`const`、`continue`、`decimal`、`default`、`delegate`、`do`、`double`、`else`、`enum`、`event`、`explicit`、`extern`、`false`、`finally`、`fixed`、`float`、`for`、`foreach`、`goto`、`if`、`implicit`、`in`、`int`、`interface`、`internal`、`is`、`lock`、`long`、`new`、`null`、`object`、`operator`、`out`、`override`、`params`、`private`、`protected`、`public`、`readonly`、`ref`、`return`、`sbyte`、`sealed`、`short`、`sizeof`、`stackalloc`、`static`、`string`、`struct`、`switch`、`this`、`throw`、`true`、`try`、`typeof`、`uint`、`ulong`、`unchecked`、`unsafe`、`ushort`、`using`、`virtual`、`void`、`volatile`、`while` 等。

1-4-2 Visual C# 命名規則

- ❖ C# 的識別字是由一個或多個字元所組成，第一個字元可以是英文字母、底線 (`_`) 或中文，其它字元可以是英文字母、底線 (`_`)、數字或中文，長度不得超過 1023 個字元。若第一個字元是底線 (`_`)，那麼必須至少包含一個英文字母、數字或中文。
- ❖ C# 會區分英文字母的大小寫，例如大寫的 `N` 和小寫的 `n` 不同。
- ❖ 由於標準類別庫或第三方類別庫幾乎都是以英文來命名，考慮到與國際接軌及社群習慣，建議不要以中文來命名。
- ❖ 建議使用有意義的英文單字和字首大寫來命名，例如 `UserName`、`StudentFirstName`，避免以單一字元命名，因為可讀性較差。
- ❖ 對於經常使用的名稱，可以使用合理的簡寫，例如以 `XML` 代替 `eXtensible Markup Language`。
- ❖ 變數的名稱建議以型別簡寫開頭，例如 `strUserName`。
- ❖ 方法的名稱建議以動詞開頭，例如 `CloseDialog`。
- ❖ 類別、結構或屬性的名稱建議以名詞開頭，例如 `UserData`。
- ❖ 介面的名稱建議以大寫字母 `I` 開頭，例如 `IComponent`。
- ❖ 事件程序的名稱建議以 `EventHandler` 結尾，例如 `MouseEventHandler`。
- ❖ 不能中斷或使用 C# 的陳述式、內建的物件 / 方法 / 列舉 / 結構 / 類別 / 事件名稱、特殊字元或空白，盡量不要使用 C# 的關鍵字，以免造成混淆。若一定要使用與關鍵字相同的識別字，或許是因為要存取以其它 .NET 語言撰寫的類別，那麼在存取該識別字時必須加上 `@` 符號做為區分，例如 C# 編譯器會將 `@class` 視為合法的識別字，而不會誤判為 `class` 關鍵字。

1-4-3 Visual C# 程式碼註解

註解可以用來記錄程式的用途與結構，C# 提供下列兩種註解符號：

- ❖ `//`：標示單行註解，可以自成一行，也可以放在一行敘述的最後，當 C# 編譯器遇到 `//` 符號時，會忽略從該 `//` 符號到該行結尾之間的敘述，不會加以執行，例如：

```
System.Console.WriteLine("Hello, world!");           // 顯示 "Hello, world!"
```

- ❖ `/* */`：標示多行註解，當 C# 編譯器遇到 `/*` 符號時，會忽略從該 `/*` 符號到 `*/` 符號之間的敘述，不會加以執行，例如：

```
/* 這是  
   多行註解 */
```

1-4-4 Visual C# 程式碼縮排

適當的縮排可以彰顯程式的邏輯與架構，提高可讀性，例如：

```
private void button1_Click(object sender, EventArgs e)  
{  
    MessageBox.Show("Hello, world!"); 在這一敘述的前面以空白鍵  
                                       或 [Tab] 鍵進行縮排  
}
```

1-4-5 Visual C# 程式碼分行與合併

C# 規定每個敘述的結尾一定要加上分號 (;)，但沒有規定換行的方式，不過，我們建議您將不同的敘述一一換行，可讀性較高。

若要將同一個敘述換行 (或許是因為太長)，可以直接按 `[Enter]` 鍵，並在最後一行的結尾加上分號；相反的，若要將多個敘述合併成一行，可以直接寫成同一行，並在每個敘述的結尾加上分號，例如 `X = 1; Y = 2; Z = 3;`。

1-5 使用 MessageBox.Show() 方法

MessageBox.Show() 方法隸屬於 `System.Windows.Forms` 命名空間，用來顯示對話方塊，裡面除了指定的訊息，還有 [確定]、[取消]、[是]、[忽略] 等按鈕，待使用者點取按鈕結束對話方塊後，就傳回代表該按鈕的數值。

MessageBox.Show() 方法有數種呼叫格式，常用的如下：

- ❖ 在對話方塊內顯示參數 *str* 所指定的字串，傳回值為 `DialogResult` 列舉，其成員包括 `OK`、`Cancel`、`Abort`、`Retry`、`Ignore`、`Yes`、`No`、`None`，分別表示使用者點取 [確定]、[取消]、[中止]、[重試]、[忽略]、[是]、[否] 按鈕及沒有點取任何按鈕。

```
public static DialogResult Show(str)
```

- ❖ 在對話方塊內顯示參數 *str1* 所指定的字串，而對話方塊的標題文字則為參數 *str2*，傳回值為 `DialogResult` 列舉。

```
public static DialogResult Show(str1, str2)
```

- ❖ 在對話方塊內顯示參數 *str1* 所指定的字串及參數 *buttons* 所指定的按鈕，而對話方塊的標題文字則為參數 *str2*，傳回值為 `DialogResult` 列舉。

```
public static DialogResult Show(str1, str2, buttons)
```

參數 *buttons* 隸屬於 `MessageBoxButtons` 列舉，其成員如下：

成員	說明
<code>OK</code>	顯示 [確定] 按鈕。
<code>OKCancel</code>	顯示 [確定]、[取消] 按鈕。
<code>AbortRetryIgnore</code>	顯示 [中止]、[重試]、[忽略] 按鈕。
<code>YesNoCancel</code>	顯示 [是]、[否]、[取消] 按鈕。
<code>YesNo</code>	顯示 [是]、[否] 按鈕。
<code>RetryCancel</code>	顯示 [重試]、[取消] 按鈕。

- ❖ 在對話方塊內顯示參數 *str1* 所指定的字串、參數 *buttons* 所指定的按鈕及參數 *icon* 所指定的圖示，而對話方塊的標題文字則為參數 *str2*，傳回值為 `DialogResult` 列舉。

```
public static DialogResult Show(str1, str2, buttons, icon)
```

參數 *icon* 隸屬於 `MessageBoxIcon` 列舉，其成員如下：

成員	說明
Error、Hand、Stop	顯示錯誤訊息圖示  。
Question	顯示問題訊息圖示  。
Exclamation、Warning	顯示警告訊息圖示  。
Information、Asterisk	顯示訊息圖示  。
None	沒有顯示圖示。

- ❖ 在對話方塊內顯示參數 *str1* 所指定的字串、參數 *buttons* 所指定的按鈕、參數 *icon* 所指定的圖示及參數 *DefaultButton* 所指定的預設按鈕，而對話方塊的標題文字則為參數 *str2*，傳回值為 `DialogResult` 列舉。

```
public static DialogResult Show(str1, str2, buttons, icon, DefaultButton)
```

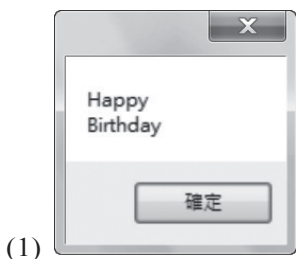
參數 *DefaultButton* 隸屬於 `MessageBoxDefaultButton` 列舉，其成員有 `Button1`、`Button2`、`Button3`，分別表示預設按鈕為對話方塊的第一、二、三個按鈕。

- ❖ 在參數 *IWin32Window* 指定的物件前面顯示對話方塊，而對話方塊內的字串及標題文字則分別為參數 *str1*、和參數 *str2*，傳回值為 `DialogResult` 列舉。

```
public static DialogResult Show(IWin32Window, str1, str2)
```

隨堂練習

撰寫兩個能夠產生如下對話方塊的敘述 (提示：換行字元為 \n 或 \r，Tab 字元為 \t)。



解答

- (1) `MessageBox.Show("Happy\nBirthday");`
- (2) `MessageBox.Show(" 大家好 \n 請多多指教 ", " 示範 ", MessageBoxButtons.OK, MessageBoxIcon.Information);`



注意

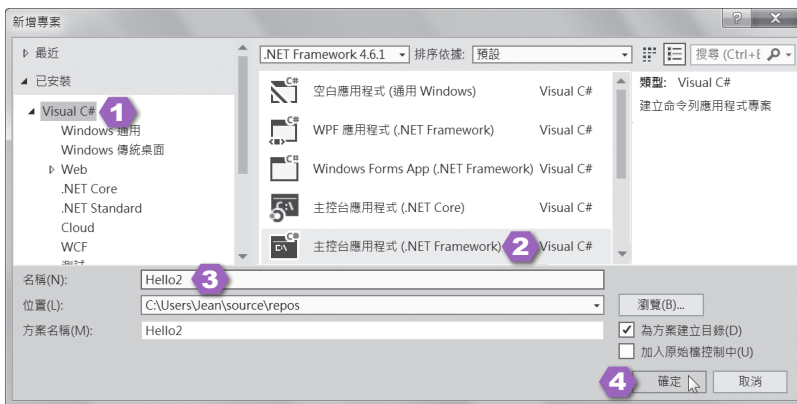
若要取得 `MessageBox.Show()` 的傳回值，可以透過類似如下的程式碼：

```
DialogResult result;           // 宣告變數 result 為 DialogResult 列舉型別
// 將 MessageBox.Show() 的傳回值指派給變數 result
result = MessageBox.Show(" 是否要登出？ ", " 詢問 ", MessageBoxButtons.YesNo);
if (result == DialogResult.Yes) // 使用 if 判斷結構檢查是否按下 [ 是 ]
{
    ...                         // 在此撰寫當按下 [ 是 ] 時所要執行的敘述
}
```

1-6 建立主控台應用程式

在前面的例子中，我們所建立的是表單應用程式，但有些情況可能不需要用到表單，此時，我們可以建立主控台應用程式，步驟如下：

1. 關閉目前開啟的方案，然後選取 [檔案] \ [新增] \ [專案]，再依照下圖操作，新增一個名稱為 **Hello2** 的主控台應用程式。



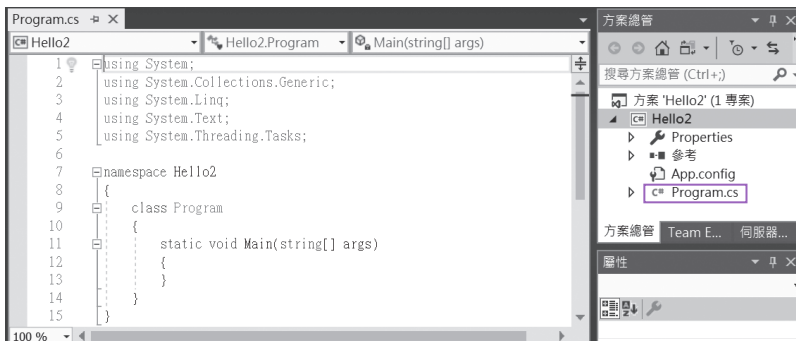
❶ 選擇 [Visual C#]

❸ 輸入專案名稱

❷ 選擇 [主控台應用程式 (.NET Framework)]

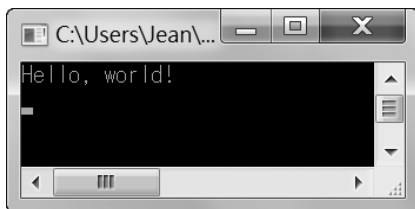
❹ 按 [確定]

2. 方案總管內出現新增的類別檔案 **Program.cs**，同時程式碼視窗內亦出現如下程式碼。



由於沒有指定命名空間與類別名稱，因此，預設的命名空間與類別分別為專案名稱和類別檔案名稱。Visual Studio 會自動匯入 System、System.Collections.Generic、System.Linq、System.Text、System.Threading.Tasks 等命名空間，而 `class Program {...}` 是宣告一個名為 Program 的類別，`static void Main(string[] args)` 是宣告一個名為 Main() 的方法，這是應用程式的起點。至於為何要宣告類別呢？因為 Visual C# 的敘述區塊並不能當作獨立的程式單元，必須放在類別內。

3. 我們可以加入程式碼，例如撰寫 Main() 方法，輸入完畢後按 [F5] 鍵建置並執行程式，結果會在執行視窗顯示 "Hello, world!"。



```
namespace Hello2
```

```
{
```

```
    class Program
```

```
    {
```

```
        static void Main(string[] args)
```

```
        {
```

```
            Console.WriteLine("Hello,world!");
```

```
            Console.ReadLine();
```

```
        }
```

```
    }
```

```
}
```

} 這兩行程式碼是我們自己撰寫的，
其它程式碼則是自動產生的。

Console 類別的 ReadLine()、WriteLine() 方法可以在執行視窗讀取一行輸入與顯示一行輸出，此例是先呼叫 WriteLine() 方法在執行視窗顯示 "Hello, world!"，為了不要立即關閉視窗，還呼叫 ReadLine() 方法等待使用者輸入，隨後只要按 [Enter] 鍵，就能關閉視窗。

1-7 使用主控台輸入 / 輸出

主控台輸入 / 輸出指的是從標準輸入 (鍵盤) 讀取使用者輸入的資料，以及將執行結果或錯誤訊息顯示在標準輸出 (執行視窗)。我們可以使用 System 命名空間的 Console 類別的 **Read()**、**ReadLine()** 方法，從標準輸入讀取一個字元和一行資料，以及使用 System 命名空間的 Console 類別的 **Write()**、**WriteLine()** 方法，在標準輸出顯示一個字元和一行資料。

舉例來說，`string Data = Console.ReadLine();` 是從標準輸入讀取一行使用者輸入的字串，然後指派給一個型別為 `string`、名稱為 `Data` 的字串變數，而 `Console.WriteLine(Data);` 則是將變數 `Data` 的值顯示在標準輸出。

在使用 **WriteLine()** 方法時，我們可以在所要輸出的字串內加上諸如 `{0}`、`{1}`、`{2}` 之類的格式化字串，`{0}` 代表的是此方法的第二個參數，`{1}` 代表的是此方法的第三個參數，依此類推，例如下面的敘述是將 `{0}` 所在的位置以第二個參數 `args.Length` 的值取代：

```
Console.WriteLine(" 您輸入的命令列字串參數的個數為 {0}", args.Length);
```

我們也可以在格式化字串中加入如下的格式化數值符號。

符號	說明	範例	顯示結果
C	以貨幣格式顯示數值	<code>Console.Write("{0:C}", 2.5);</code>	NT\$2.50
D	以十進位顯示數值	<code>Console.Write("{0:D5}", 25);</code>	00025
E	以科學記號顯示數值	<code>Console.Write("{0:E}", 250000);</code>	2.500000E+005
F	以小數點後面固定位數顯示數值，不足的位數補 0	<code>Console.Write("{0:F2}", 25);</code>	25.00
G	以一般格式顯示數值	<code>Console.Write("{0:G}", 2.5);</code>	2.5
N	以千分位格式顯示數值	<code>Console.Write("{0:N}", 2500000);</code>	2,500,000.00
X	以十六進位顯示數值	<code>Console.Write("{0:X}", 250);</code>	FA

一、選擇題

- () 1. 若要在表單上插入按鈕，可以使用工具箱的哪個控制項？
A. TextBox B. Button C. Picture D. CheckBox
- () 2. 若要修改表單的標題，可以使用哪個屬性？
A. Title B. Text C. Tag D. Location
- () 3. 下列哪個特殊字元可以用來標示單行註解？
A. // B. & C. ; D. :
- () 4. 下列哪個快速鍵可以用來執行程式？
A. [F1] B. [F3] C. [F5] D. [F10]
- () 5. Visual C# 可以使用哪個陳述式宣告命名空間？
A. class B. module C. static D. namespace
- () 6. 下列哪個關鍵字可以用來匯入命名空間？
A. exports B. imports C. namespace D. using
- () 7. 下列何者為應用程式的進入點？
A. Start() B. Load() C. Main() D. Page_Load()
- () 8. C# 變數可以使用下列何者做為命名開頭？
A. _ B. !
C. 阿拉伯數字 D. #
- () 9. 若要顯示對話方塊，可以呼叫 MessageBox 類別的哪個方法？
A. Show() B. Equals()
C. Write() D. Print()
- () 10. 下列哪個字元表示 Tab ？
A. '\0' B. '\n'
C. '\t' D. '\r'

- () 11. System.Console 類別的哪個方法可以在主控台讀取一行？

- A. Read() B. ReadLine()
C. Write() D. WriteLine()

- () 12. MessageBox.Show() 方法的哪個傳回值代表使用者點取「忽略」？

- A. Retry
B. Ignore
C. Cancel
D. Abort

二、練習題

1. 撰寫一個 Visual C# 程式，令其執行結果如下。



2. 試問，下面的 Visual C# 程式碼有沒有錯誤？若有的話，那是什麼錯誤？又該如何更正呢？

```
class Program
{
    static void main(string[] args)
    {
        Console.WriteLine("Hello, world!")
    }
}
```