

# C#

## (Environment: Visual Studio)

RoJeA

# Introduction to C#

- Introduction to C# Windows Forms Applications
- <https://www.geeksforgeeks.org/introduction-to-c-sharp-windows-forms-applications/>
- **WHY DO YOU TEACH CONSOLE APPLICATIONS INSTEAD OF DRAG AND DROP IN WINFORMS?**
- <https://iamtimcorey.com/teach-console-applications-instead-drag-drop-winforms/>

- Declaration
- Array Sort
- 串列(List)
- Convert & Math Round(Digits)
- DateTime 格式
- C# read txt
- String修剪
- C# log file

- 取得路徑
- DirectoryInfo.GetFiles Method
- C# & SQL
- C# using R to Calculate & Data Mining
- Web Crawler(1)
- C#實現的三種方式實現模擬鍵盤按鍵
- 如何使用 VS Code 建立 .NET Core 開發環境？

# Declaration

- `int a = 100;`      // 整數
- `double b = 1.12;`    // 浮點數，可決定是否增加後置字元 `d` 或 `D`
- `float c = 3.5F;`    // 浮點數。容易被視為 `double`，因此需後置字元 `f` 或 `F` 初始化 `float` 變數
- `char d = 'A';`      // 單一字元
- `string e = "World!";` // 字串
- `bool f = true, g = false;` // 布林 (邏輯)
- `public static string dd;` // 記錄用之宣告

# Declaration

- `int[] ABC = new int[5]`
- `string[] Words=new string[3]`
- `int[,] array = new int[4, 2];`
- ※建立[陣列]時未設定初始值，如[數值資料型別]預設值為[零]；  
[字串資料型別]預設為[**null**]；[布林值資料型別]預設為[**false**]。

# Array Sort

- 陣列宣告、排序、反轉

- 陣列的排序：此指定一維物件由[小而大]作遞增排序。

方法一、**Array.Sort(陣列物件);**

方法二、**Array.Sort(陣列物件1,陣列物件2);**

- 陣列的反轉：由[大而小]作遞減排列。

方法、**Array.Reverse(陣列物件);**

※陣列必須先作(Sort)排列後，在作(Reverse)反轉。

※假若同時有兩個相關的陣列A和B，若以A陣列為基準由大到小做排序，其相關陣列需要同時反轉。寫法如下：

```
Array.Sort(A,B);           //將A陣列先做由小而大排序，B陣列亦跟著修改。  
Array.Reverse(A);          //將A陣列先做反轉，變成由大而小排序。  
Array.Reverse(B);          //將B陣列做反轉。
```

# 串列(List)

- Basic

`List<T>`建立:須預為串列裡要存放的項目指明『物件』或『值的型別』。( `<T>` 可以被取代為某種型別，所以 `List<string>` 代表由 `string` 構成的串列。)

- 新增項目

- `// Create list`
- `var myList = new List<string>();`
  
- `// Add items to the list`
- `myList.Add("item1");`
- `myList.Add("item2");`
  
- `// Convert to array`
- `var myArray = myList.ToArray();`



# Convert & Math Round(Digits)

- `Tv = Convert.ToDouble(T);`
- `int numVal = Int32.Parse("-105");`
- `DateTime dt1 = Convert.ToDateTime("2007-8-1");`
- `A=Math.Round(kwrtv[2], 4)`

# DateTime 格式

- DateTime.Now.ToShortTimeString()  
DateTime dt = DateTime.Now;  
dt.ToString();//2005-11-5 13:21:25  
dtToFileTime().ToString();//127756416859912816  
dtToFileTimeUtc().ToString();//127756704859912816  
dt.ToLocalTime().ToString();//2005-11-5 21:21:25  
dt.ToLongDateString().ToString();//2005年11月5日  
dt.ToLongTimeString().ToString();//13:21:25  
dt.ToOADate().ToString();//38661.5565508218  
dt.ToShortDateString().ToString();//2005-11-5  
dt.ToShortTimeString().ToString();//13:21  
dt.ToUniversalTime().ToString();//2005-11-5 5:21:25  
dt.Year.ToString();//2005  
dt.Date.ToString();//2005-11-5 0:00:00

- dt.DayOfWeek.ToString();//Saturday  
dt.DayOfYear.ToString();//309  
dt.Hour.ToString();//13  
dt.Millisecond.ToString();//441  
dt.Minute.ToString();//30  
dt.Month.ToString();//11  
dt.Second.ToString();//28  
dt.Ticks.ToString();//632667942284412864  
dt.TimeOfDay.ToString();//13:30:28.4412864  
dt.ToString();//2005-11-5 13:47:04

- dt.AddYears(1).ToString();//2006-11-5 13:47:04  
dt.AddDays(1.1).ToString();//2005-11-6 16:11:04  
dt.AddHours(1.1).ToString();//2005-11-5 14:53:04  
dt.AddMilliseconds(1.1).ToString();//2005-11-5 13:47:04  
dt.AddMonths(1).ToString();//2005-12-5 13:47:04  
dt.AddSeconds(1.1).ToString();//2005-11-5 13:47:05  
dt.AddMinutes(1.1).ToString();//2005-11-5 13:48:10  
dt.AddTicks(1000).ToString();//2005-11-5 13:47:04  
dt.CompareTo(dt).ToString();//0  
dt.Add(?).ToString();//問號為一個時間段  
dt.Equals("2005-11-6 16:11:04").ToString();//False  
dt.Equals(dt).ToString();//True  
dt.GetHashCode().ToString();//1474088234  
dt.GetType().ToString();//System.DateTime  
dt.GetTypeCode().ToString();//DateTime

# C# read txt

- using System.IO;
- var lastLine = File.ReadLines(@"C:\Users\608L\Desktop\download.txt").Last(); //最後一行
- label1.Text = lastLine;
- string y61, y62;
- y61 = lastLine.Substring(18, 4); //第18個位置，取4個
- y62 = lastLine.Substring(40, 2);
- label2.Text = y61;
- label3.Text = y62;

# String修剪

- `int typeplace = dd.LastIndexOf(".");`
- `string TYPE = dd.Substring(typeplace);`//獲得上傳的圖片的字尾名
- `string outDirPath = Server.MapPath("~/file/");`
- `string inFilePath = outDirPath + dd;`
- `string filename = dd.TrimEnd(TYPE.ToCharArray());`//刪除字符串頭部及尾部出現的a或b或c或d字符，刪除的過程直到碰到一個既不是a也不是b也不是c也不是d的字符才結束。
- `string[] filename = dd.Split('.');`//字串尾部的.副檔名字元
  - `//filename[0]:X000001 filename[1]:xlsx`

# C# Log File

- `/// FileStream寫入用法`
- `private void FileStreamWriteFile(string a, string b)`
- `{`
- `string strAppPath = Request.PhysicalApplicationPath;`
- `FileStream fsFile = new FileStream(strAppPath + "LogFile//log.csv",`  
`FileMode.OpenOrCreate);`
- `StreamWriter swWriter = new StreamWriter(fsFile);`
- `//將指針設定起始位置`
- `fsFile.Seek(0, SeekOrigin.End);`
- `//寫入數據`
- `swWriter.WriteLine("{0}," + a + b, DateTime.Now.ToOADate().ToString());`
- `swWriter.Close();`
- `}`

# 取得路徑

- `string outDirPath = System.Windows.Forms.Application.StartupPath + @"..\..\\" + @"file\";` //往上兩層，往下一層(file) <https://dotblogs.com.tw/supershowwei/2017/01/22/004746>

- 摘要:如何取得目前程式執行的根目錄
  - 若在 asp.net 裡想取得根目錄的實體位置,可以寫成  
`string path = Server.MapPath("/");`

若在 Windows Forms 可以寫成  
`string path = Application.StartupPath ;`

若在 Console Application 可以寫成  
`string path=System.AppDomain.CurrentDomain.BaseDirectory;`

如果您想寫一支 dll 專案,供上述專案類型參考並叫用,而您想在 dll 取得目前該專案的根目錄,則可以寫成  
`string path=System.AppDomain.CurrentDomain.BaseDirectory;`

是可以在上述三種專案裡同時正確執行。

<https://dotblogs.com.tw/chiajung/archive/2009/11/04/11415.aspx>



# DirectoryInfo.GetFiles Method

<a href="#"><u>GetFiles(String, SearchOption)</u></a>	從目前目錄傳回符合指定搜尋模式的檔案清單，並使用值來判斷是否搜尋子目錄。
<a href="#"><u>GetFiles()</u></a>	從目前的目錄傳回檔案清單。
<a href="#"><u>GetFiles(String)</u></a>	從目前目錄傳回符合指定之搜尋模式的檔案清單。

<https://docs.microsoft.com/zh-tw/dotnet/api/system.io.directoryinfo.getfiles?view=netframework-4.8>

# 計算2個日期之間的天數差

```
DateTime dt1 = Convert.ToDateTime("2007-8-1");  
DateTime dt2 = Convert.ToDateTime("2007-8-15");  
TimeSpan span = dt2.Subtract(dt1);  
int dayDiff = span.Days + 1;  
計算某年某月的天數
```

```
-----  
int days = DateTime.DaysInMonth(2007, 8);  
days = 31;  
給日期增加一天、減少一天
```

```
-----  
DateTime dt = DateTime.Now;  
dt.AddDays(1); //增加一天  
dt.AddDays(-1); //減少一天  
其它年份方法類似...  
Oracle SQL裡轉換日期函數
```

```
-----  
to_date("2007-6-6",YYYY-MM-DD");  
to_date("2007/6/6",yyyy/mm/dd");  
如下一組數據,如何查找表裡包含9月份的記錄:  
CGGC_STRATDATE CGGC_ENDDATE
```

```
=====
```

2007-8-4	2007-9-5
2007-9-5	2007-9-20
2007-9-22	2007-10-5

```
SELECT * FROM TABLE  
(TO_DATE(2007/9/1,yyyy/mm/dd) BETWEEN CGGC_STRATDATE  
AND CGGC_ENDDATE OR CGGC_STRATDATE >=TO_DATE(2007/9/1,yyyy/mm/dd)  
AND CGGC_ENDDATE<=TO_DATE(2007/9/30,yyyy/mm/dd) "  
OR TO_DATE(2007/9/30,yyyy/mm/dd) BETWEEN CGGC_STRATDATE  
AND CGGC_ENDDATE) ORDER BY CGGC_STRATDATE ASC
```

# C# & SQL

- `using System.Data.SqlClient;`
- `using System.Linq;`
- `using (SqlConnection connection = new SqlConnection(`
- `connectionString))`
- `{`
- `SqlCommand command = new SqlCommand(queryString, connection);`
- `command.Connection.Open();`
- `// Do work here; connection closed on following line.`
- `command.ExecuteNonQuery();`
- `command.Connection.Close();`
- `}`

# C# & SQL INSERT

- `using System.Data.SqlClient;`
- `using (SqlConnection con = new SqlConnection("Data Source = 140.124.47.153; Initial Catalog = new_ASUS; Persist Security Info = True; User ID = remote test; Password = 654321"))`
- `{`
- `con.Open();//open connection`
- `SqlCommand insertCommand = new SqlCommand("INSERT INTO [EightYearWeather].[dbo].[鞍部] ([DateTime],[StnPres]) VALUES (@value0,@value1)", con);`
- `insertCommand.Parameters.AddWithValue("@value0", TD + " " + hr.ToString() + ":00:00");`
- `insertCommand.Parameters.AddWithValue("@value1", value_output[hr, 1, D]);“`
- `insertCommand.ExecuteNonQuery();`
- `con.Close();`
- `}`

# C# & SQL TRY&CATCH(not implemented yet)

- `using System.Data.SqlClient;`
- `string connetionString = null;`
- `SqlConnection cnn ;`
- `connetionString = "Data Source=ServerName;Initial Catalog=DatabaseName;User ID=UserName;Password=Password"`
- `cnn = new SqlConnection(connetionString);`
- `try`
- `{`
- `cnn.Open();`
- `MessageBox.Show ("Connection Open ! ");`
- `cnn.Close();`
- `}`
- `catch (Exception ex)`
- `{`
- `MessageBox.Show("Can not open connection ! ");`
- `}`

# Application Exit

- `this.Close();`
- `Environment.Exit(Environment.ExitCode);`
- OR
  - `//Application.Exit();`

# C# & SQL To Gridview

- //MsSql connection
- SqlConnection conn = new SqlConnection("Data Source=IP;Initial Catalog=DB;User ID=xxx;Password=xxx");
- conn.Open();
- SqlCommand cmd = **new** SqlCommand("Select odh\_no as 訂單編號, odh\_cm as 客戶編號 From odh where To\_JEAN = 'T'", conn);
- DataTable dataTable = new DataTable();
- SqlDataAdapter da = new SqlDataAdapter(cmd);
- da.Fill(dataTable);
- GridView1.DataSource = dataTable; //告訴GridView資料來源為誰
- GridView1.DataBind();//綁定
- conn.Close(); //連線關閉

# C# using R to Calculate & Data Mining

- NuGet 封裝管理員:Install-Package R.NET.Community -Version 1.7.0



# C# using R to Calculate & Data Mining

- using RDotNet;
- REngine.SetEnvironmentVariables();
- REngine engine = REngine.GetInstance();
- engine.Evaluate("library(e1071)");
- engine.Evaluate("library(RODBC)");
- engine.Evaluate("con = odbcConnect('new\_ASUS',uid='remote test', pwd='654321')");
- engine.Evaluate("rt100 <- sqlQuery(con,'SELECT \* FROM[new\_ASUS].[dbo].[TRHCL] WHERE [Tdb]> " + TdbL + " and [Tdb] < " + TdbH + " and [RH] > " + RH1L + " and [RH] < " + RH1H + "')");
- DataFrame dataset = engine.GetSymbol("rt100").AsDataFrame();
- NumericVector Res = engine.Evaluate("test=sample(1:nrow(rt100), size=0.2\*nrow(rt100))").AsNumeric();
- engine.Evaluate("train=rt100[-test,]");
- CharacterVector Result00 = engine.Evaluate("rf=svm(CL ~ ., data=train,importance=T,proximity=T)").AsCharacter();
- CharacterVector Result01 = engine.Evaluate("ZZ <- data.frame(Tdb=c(" + Tdb + "),RH=c(" + RH1 + "))").AsCharacter();
- CharacterVector Result02 = engine.Evaluate("pred=predict(rf,newdata=ZZ)").AsCharacter();
- CharacterVector Result03 = engine.Evaluate("pred").AsCharacter();

# Web Crawler(1)

- using System.Net;
- string TD = EightYearAgo.AddDays(D).ToString("yyyy-MM-dd");
- url[D] = "http://e-service.cwb.gov.tw/HistoryDataQuery/DayDataController.do?command=viewMain&station=466910&stname=%25E9%259E%258D%25E9%2583%25A8&datepicker=" + TD;
- //讀取氣象站網頁
- WebRequest myRequest = WebRequest.Create(@url[D]);
- myRequest.Method = "GET";//Method選擇GET
- WebResponse myResponse = myRequest.GetResponse();//取得WebRequest  
的回覆
- StreamReader sr = new  
StreamReader(myResponse.GetResponseStream());//Streamreader讀取回覆
- string result = sr.ReadToEnd();//將全文轉成string
- sr.Close();//關掉StreamReader

# Web Crawler(1)

- //開始抓取需要資料
- int mid, end;
- int hour = 25, size = 15;
- int[] star = new int[hour]; //建立計算小時矩陣
- int[,] length = new int[hour, size]; //建立儲存字串長度矩陣
- string[,] value\_get = new string[hour, size]; //建立儲存數值矩陣
- string[,,,] value\_output = new string[hour, size, Diffday]; //建立輸出數值矩陣
- end = result.Length; //計算總字串長度

# Web Crawler(1)

- for (int hr = 1; hr <= 24; hr++)//逐小時抓取
- {
- star[hr] = result.IndexOf("<td nowrap>" + hr);//蒐尋起使點
- value\_get[hr, 0] = result.Substring(star[hr] + 1, end - star[hr] - 1);
- for (int i = 1; i <= 14; i++)//抓取14項資料(測站氣壓~能見度)
- {
- length[hr, i] = value\_get[hr, i - 1].IndexOf("<td>");
- value\_get[hr, i] = value\_get[hr, i - 1].Substring(length[hr, i] + 1,
- value\_get[hr, i - 1].Length - length[hr, i] - 1);
- mid = value\_get[hr, i].IndexOf("&");
- value\_output[hr, i, D] = value\_get[hr, i].Substring(3, mid - 3);
- }
- }

# C#实现的三种方式实现模拟键盘按键

- 组合键：Ctrl = ^ 、 Shift = + 、 Alt = %  
模拟按键： A
- ```
private void button1_Click(object sender, EventArgs e) {  
    textBox1.Focus();  
    SendKeys.Send("{A}");  
}
```

# 如何使用 VS Code 建立 .NET Core 開發環境？

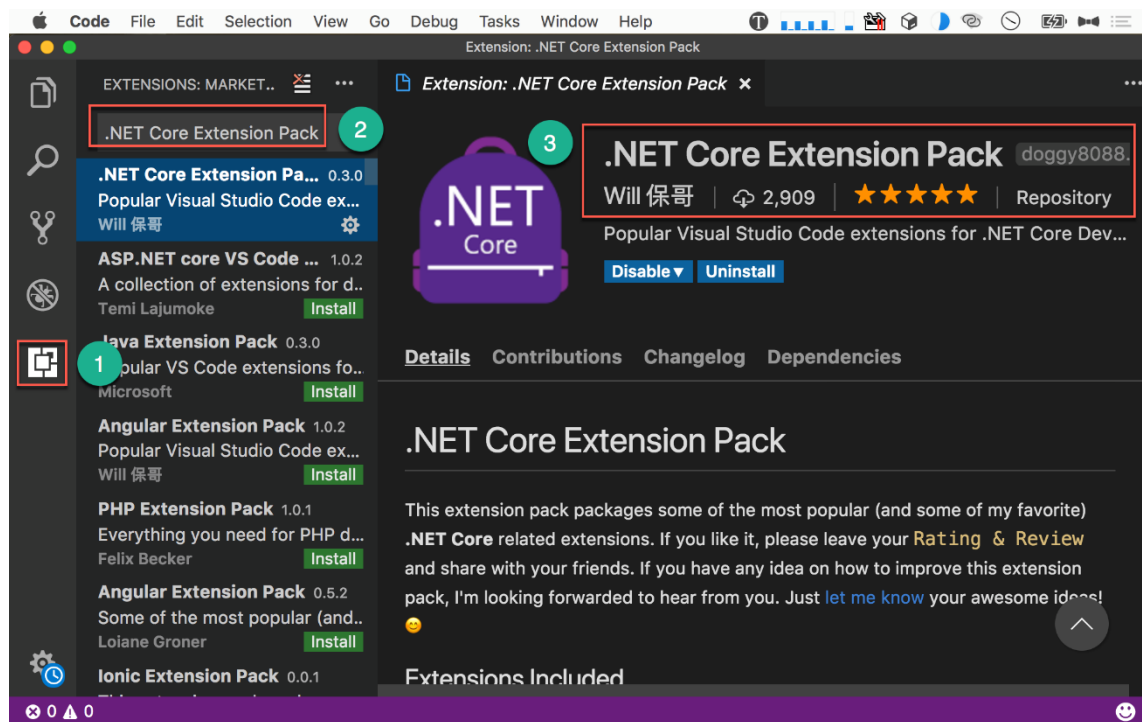
## Extension

1. 按下左側 **Extension** icon

2. 輸入 **.NET Core Extension Pack**

3. 選擇 Will 保哥所整理 .NET Core Extension Pack

由於 VS Code 為 open source project，當然可以自行選擇各種 extension 使用，此為保哥所整理的好用 extension，基本上安裝保哥的版本就已經足夠使用



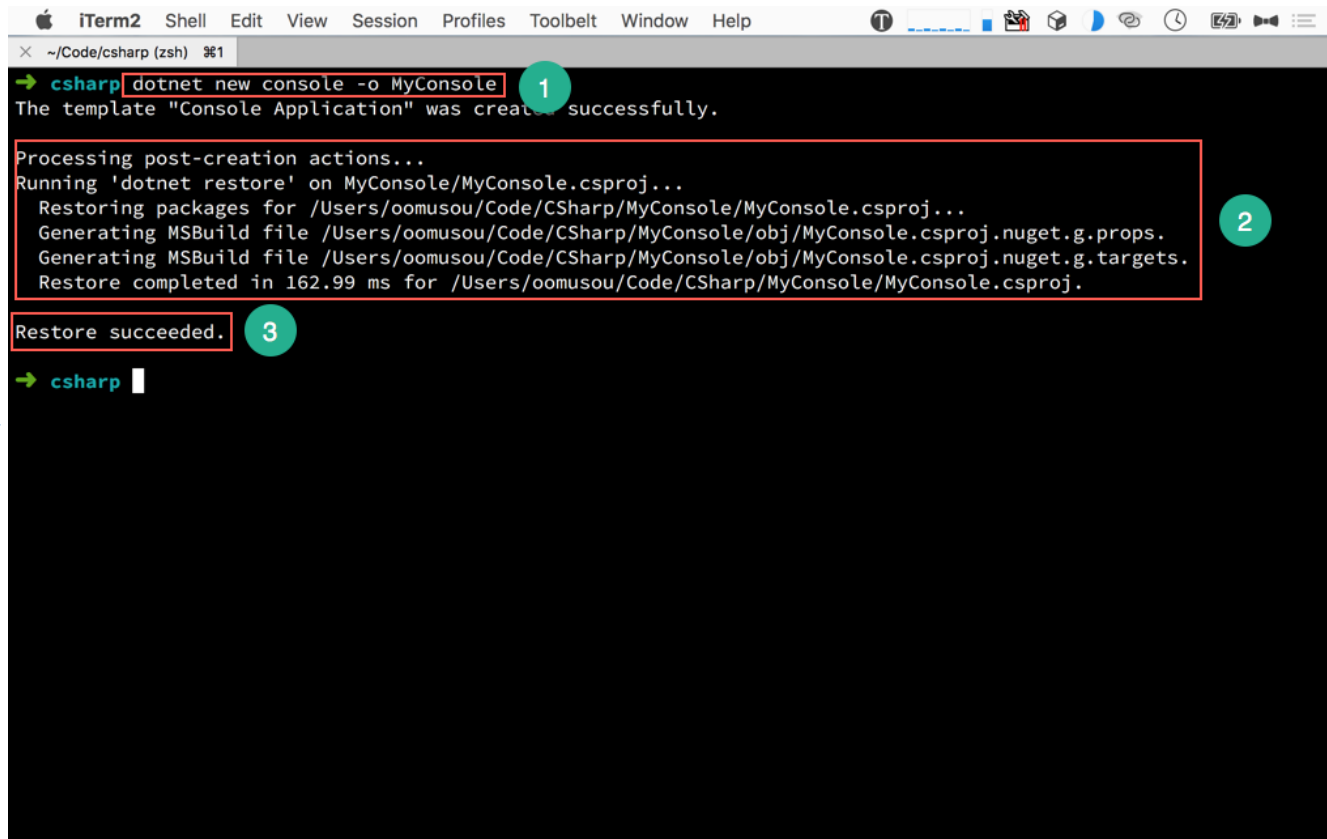
# 如何使用 VS Code 建立 .NET Core 開發環境？

## Hello World

### 建立專案

\$ dotnet new console -o MyConsole

- **new** : 建立新專案
- **console** : 建立 console 類型專案
- **-o** : o output，表建立在 **MyConsole** 目錄下



```
iTerm2 Shell Edit View Session Profiles Toolbelt Window Help
~/Code/csharp (zsh) %1
→ csharp dotnet new console -o MyConsole 1
The template "Console Application" was created successfully.

Processing post-creation actions...
Running 'dotnet restore' on MyConsole/MyConsole.csproj...
Restoring packages for /Users/oomusou/Code/CSharp/MyConsole/MyConsole.csproj...
Generating MSBuild file /Users/oomusou/Code/CSharp/MyConsole/obj/MyConsole.csproj.nuget.g.props.
Generating MSBuild file /Users/oomusou/Code/CSharp/MyConsole/obj/MyConsole.csproj.nuget.g.targets.
Restore completed in 162.99 ms for /Users/oomusou/Code/CSharp/MyConsole/MyConsole.csproj. 2

Restore succeeded. 3
→ csharp
```

1. 輸入 **dotnet new console -o MyConsole** 將 console 類型專案建立在 **MyConsole** 目錄下
2. .NET Core SDK 開始建立專案所需的檔案
3. 自動下載所需要的 NuGet package

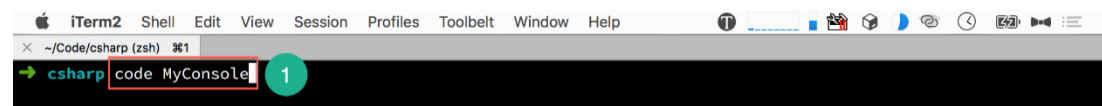
# 如何使用 VS Code 建立 .NET Core 開發環境？

## VS Code 開啟專案

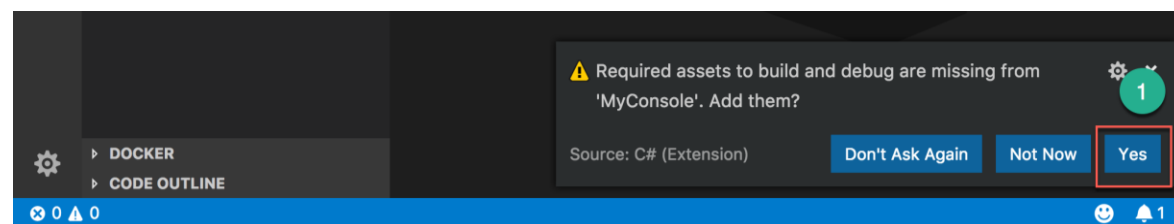
使用 `code` 執行 VS Code，後面接 開啟目錄名稱。

- `$ code MyConsole`

1. 輸入 `code MyConsole` 要求 VS Code 直接開啟 `MyConsole` 目錄

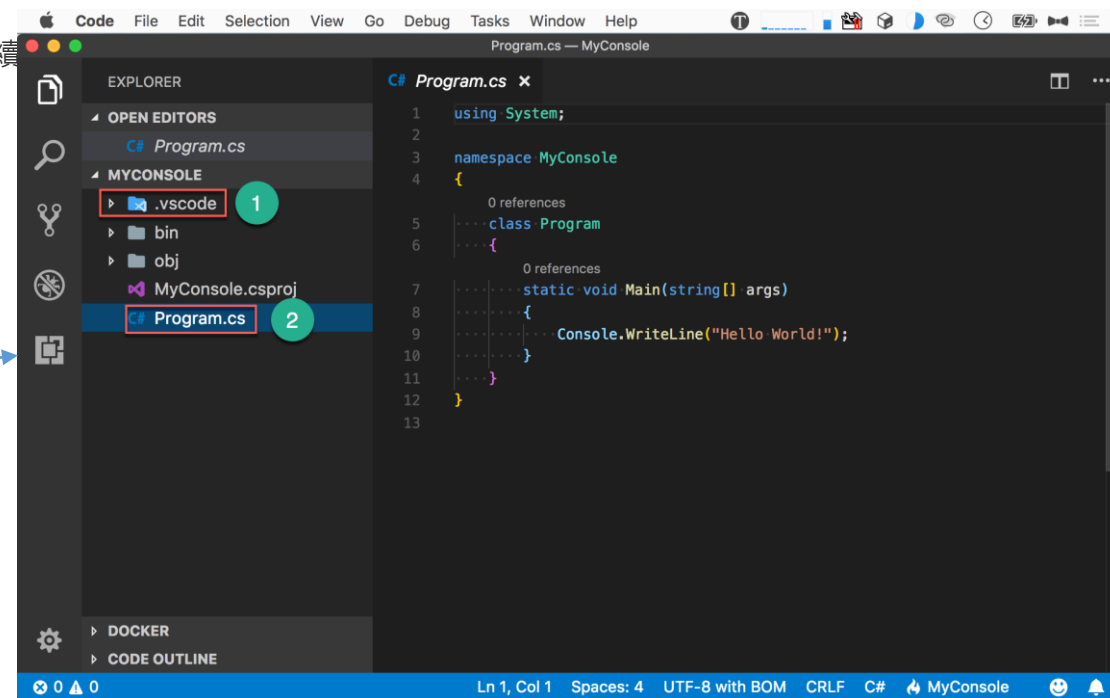


2. 第一次使用 VS Code 開啟 .NET Core 專案，會要求建立 `.vscode` 設定檔目錄，按 `Yes` 繼續



3. `.vscode` 被 VS Code 自動建立

4. 點選 `Program.cs`，VS Code 已經能辨識出 C#，並支持語法變色與 Intellisense





# 如何使用 VS Code 建立 .NET Core 開發環境？

## • 編譯 .NET Core

- `$ dotnet build`

### • **build** : 編譯專案

1. 按熱鍵 `Ctrl + `` 開啟內建的 terminal，輸入 `dotnet build` 編譯目前專案
2. .NET Core SDK 將編譯成 `MyConsole.dll`，將路徑複製下來

## 執行 .NET Core

使用 `dotnet` 執行 `dll`。

- `$ dotnet /Users/oomusou/Code/CSharp/MyConsole/bin/Debug/netcoreapp2.0/MyConsole.dll`

1. 輸入 `dotnet`，並將剛剛複製的 `dll` 路徑貼上
2. 顯示 `Hello World!`

