SFML Intro

Yu-Hsuan Chen

2018 May

Outline

- SFML Introduction
- SFML + Visual Studio
  - Visual Studio 2010 + SFML 2.3.2
  - Visual Studio 2017 + SFML 2.4.2 + CMake
- Your First SFML Program （施工中）
- Block Game with SFML（施工中）

SFML Introduction

- SFML = simple and fast multimedia library
- 跨平台、相容多語言的圖形介面
  主要為C++、.NET，社群亦有Java與Python的版本
- https://sfml-dev.org
- 目前最新的版本：2.5.0（2018/5）
  相容Visual Studio 2017

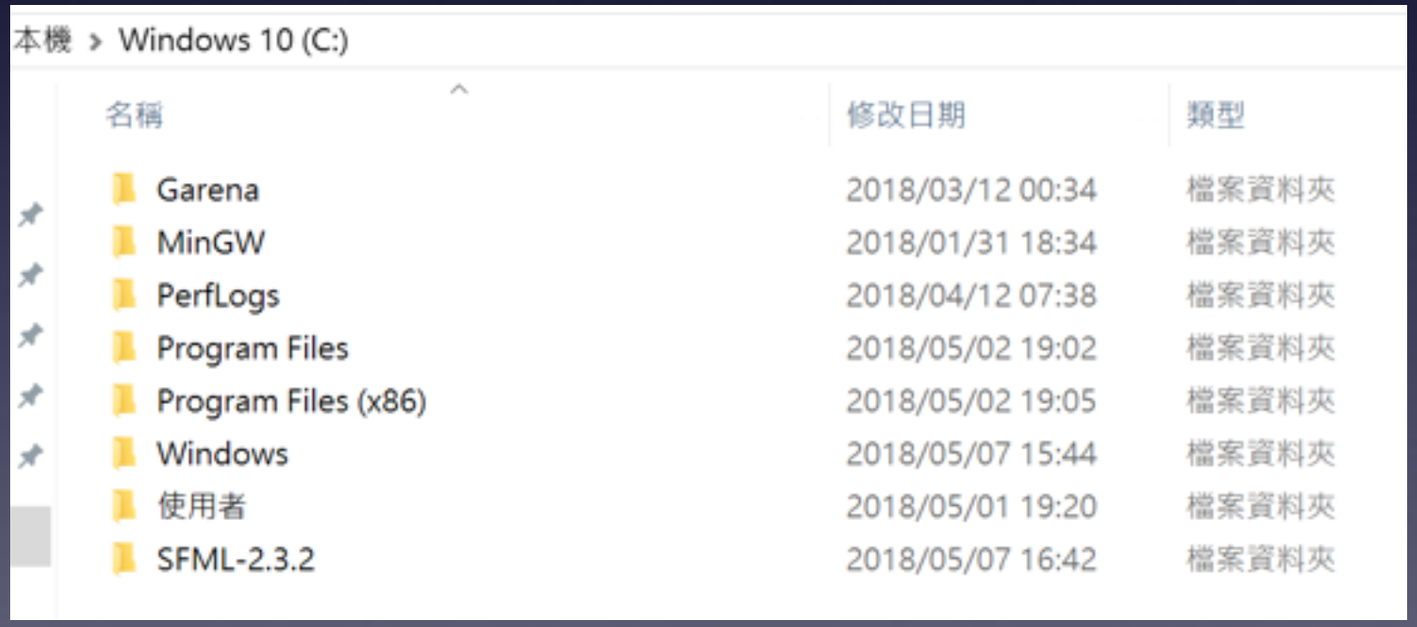
SFML 2.3.2
x
Visual Studio 2010

VC10(2010) 環境部署 1

- 官方載點：https://www.sfml-dev.org/download/sfml/2.3.2/

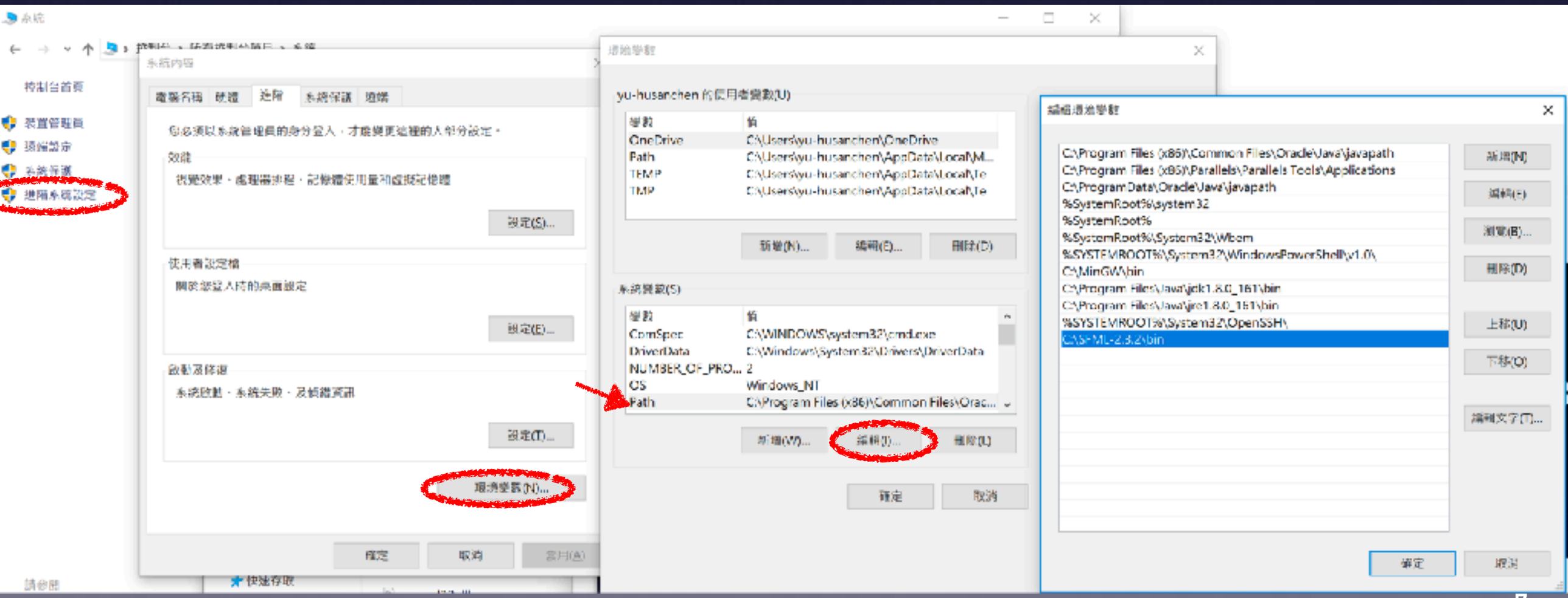
VC10(2010) 環境部署 2

- 解壓縮之後貼到適當的地方，本例子是放在C槽根目錄下



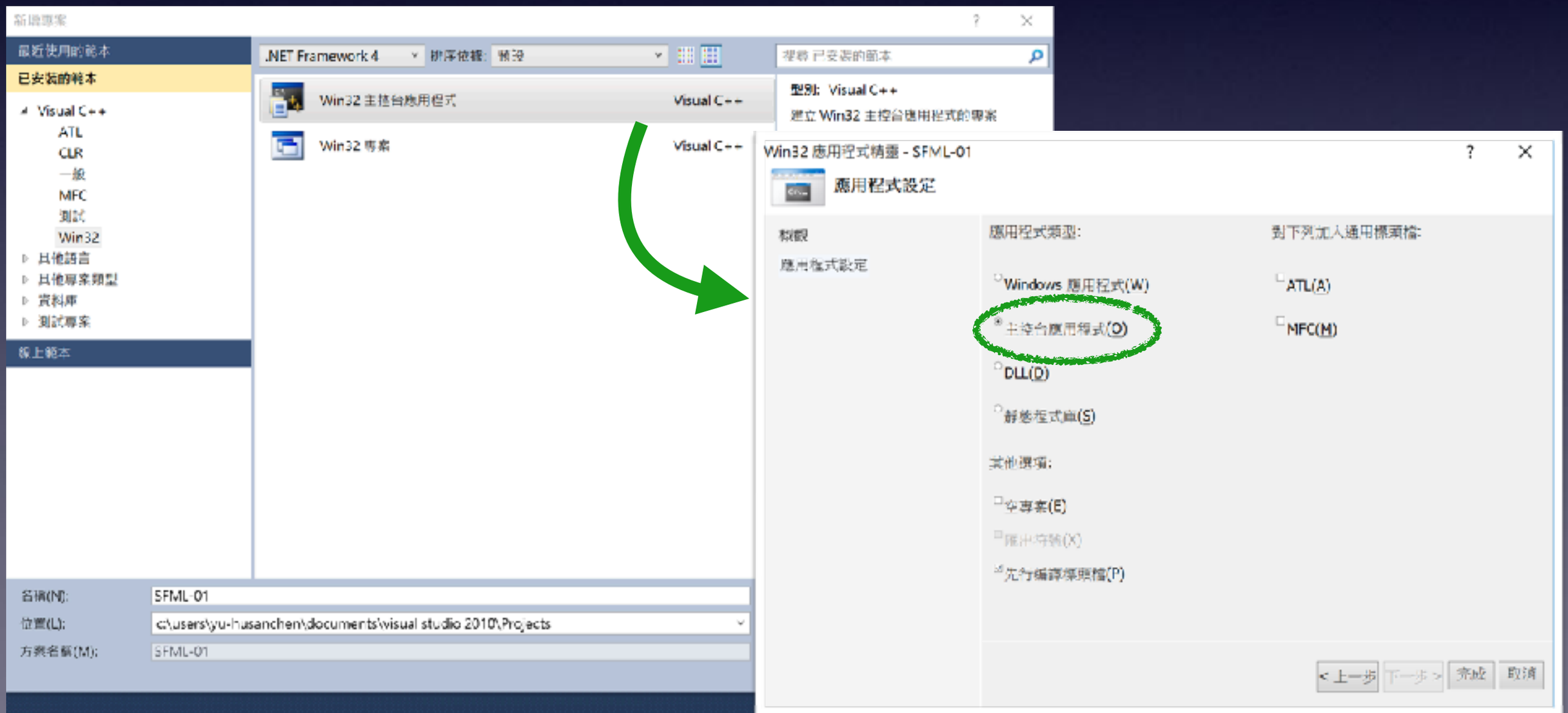
VC10(2010) 環境部署 3

- 先設定環境變數: 控制台>系統>進階系統設定
  加入 C:\SFML-2.3.2\bin
  忘記這一步會造成執行階段缺少dll而無法運作



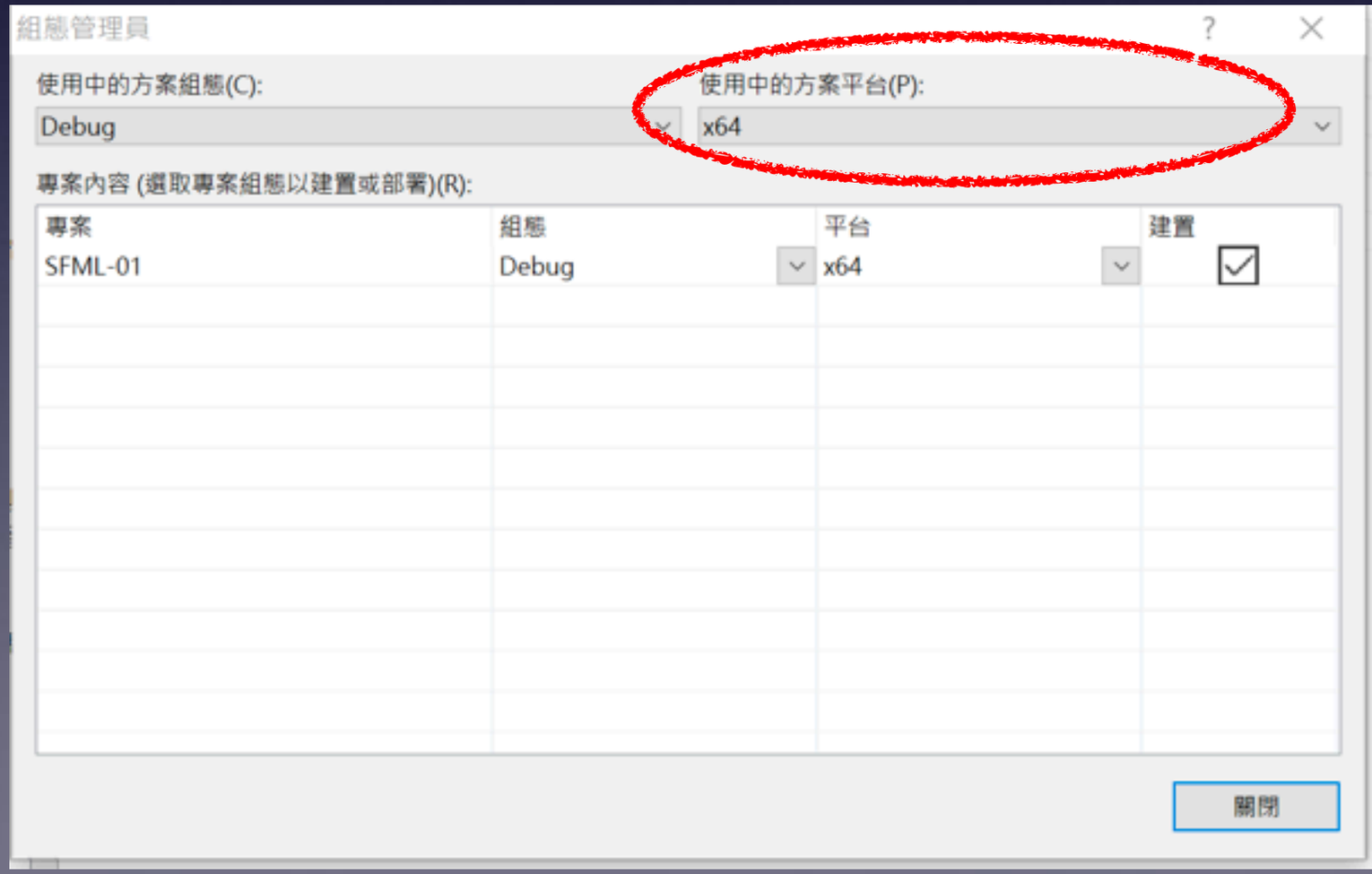
VC10(2010) 環境部署 4

- 啟動Visual Studio，並且建立一個Win32主控台應用程式專案



VC10(2010) 環境部署 5

- 必須先引入SFML相關的標頭檔。
- 專案>屬性 開啟專案屬性頁
- 點擊組態管理員，新增一個x64平台



VC10(2010) 環境部署 6

- 組態屬性>C/C++>一般: 其他Include目錄
  （組態：所有組態）
  加入 C:\SFML-2.3.2\include

- 組態屬性>連結器>一般: 其他程式庫目錄
  （組態：所有組態）
  加入 C:\SFML-2.3.2\lib

- 組態屬性>連結器>輸入: 其他相依性 （組態：Debug）
  加入
  sfml-graphics-d.lib; sfml-window-d.lib; sfml-system-d.lib; sfml-audio-d.lib; sfml-network-d.lib; kernel32.lib; user32.lib; gdi32.lib; winspool.lib; comdlg32.lib; advapi32.lib; shell32.lib; ole32.lib; oleaut32.lib; uuid.lib; odbc32.lib; odbccp32.lib; %(AdditionalDependencies)

- 組態屬性>連結器>輸入: 其他相依性 （組態：Release）
  加入
  sfml-graphics.lib;sfml-window.lib; sfml-system.lib; sfml-audio.lib; sfml-network.lib;kernel32.lib; user32.lib; gdi32.lib; winspool.lib; comdlg32.lib; advapi32.lib;shell32.lib; ole32.lib; oleaut32.lib; uuid.lib; odbc32.lib; odbccp32.lib; %(AdditionalDependencies)

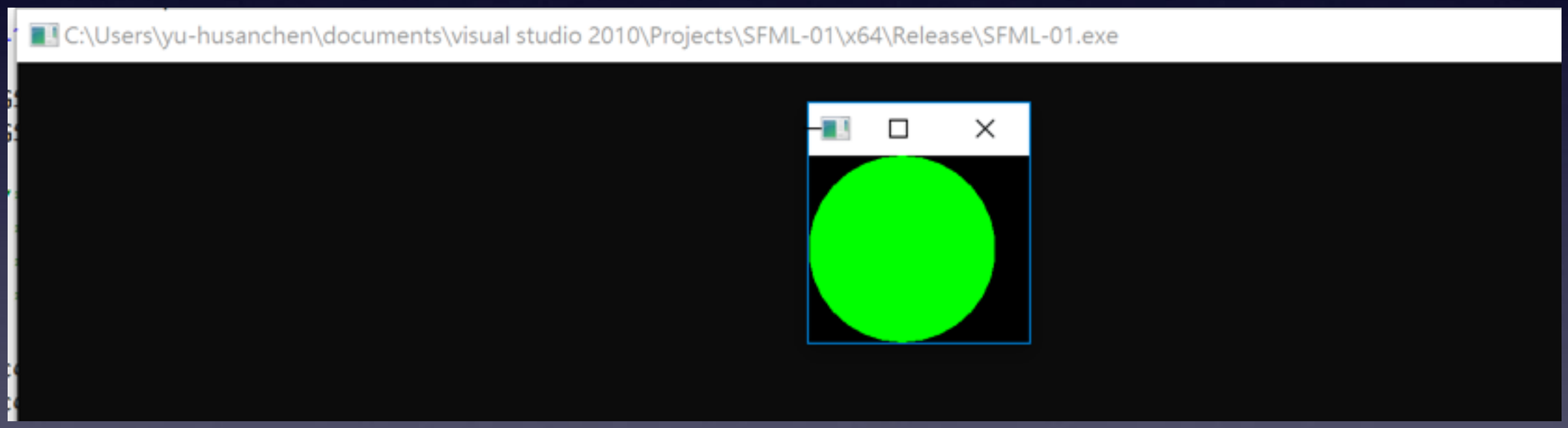
- 留意不要把原來的東西給複寫了

測試程式碼

```
#include <SFML/Graphics.hpp>
int main()
{
    sf::RenderWindow window(sf::VideoMode(200, 200), "SFML works!");
    sf::CircleShape shape(100.f);
    shape.setFillColor(sf::Color::Green);
    while (window.isOpen())
    {
        sf::Event event;
        while (window.pollEvent(event))
        {
            if (event.type == sf::Event::Closed)
                window.close();
        }

        window.clear();
        window.draw(shape);
        window.display();
    }
    return 0;
}
```

執行結果


C:\Users\yu-husanchen\documents\visual studio 2010\Projects\SFML-01\x64\Release\SFML-01.exe

CMake & SFML 2.4.2

Build library in VS2017

附註：撰寫本篇時還停留在2.4.2，最新的2.5.0已經在2018.5.10發布，
可以直接相容Visual Studio 2017，可依照前述2010的做法部署環境

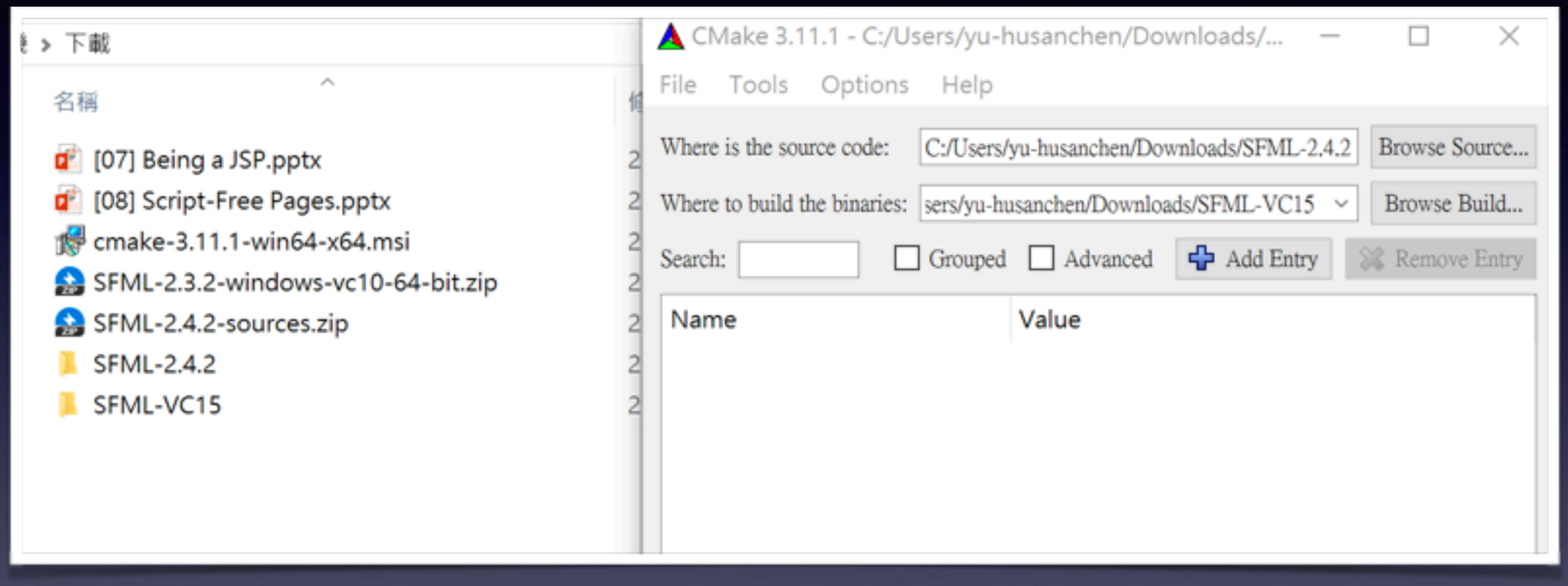
CMake in SFML 1

Download SFML 2.4.2

Windows	Visual C++ 11 (2012) - 32-bit	Download | 16.7 MB	Visual C++ 11 (2012) - 64-bit	Download | 18.5 MB
	Visual C++ 12 (2013) - 32-bit	Download | 16.1 MB	Visual C++ 12 (2013) - 64-bit	Download | 17.8 MB
	Visual C++ 14 (2015) - 32-bit	Download | 16.0 MB	Visual C++ 14 (2015) - 64-bit	Download | 17.6 MB
	GCC 4.9.2 TDM (SJLJ) - 32-bit	Download | 13.8 MB	GCC 4.9.2 TDM (SJLJ) - 64-bit	Download | 15.8 MB
	GCC 6.1.0 MinGW (DW2) - 32-bit	Download | 15.3 MB	GCC 6.1.0 MinGW (SEH) - 64-bit	Download | 16.2 MB
	On Windows, choosing 32 or 64-bit libraries should be based on which platform you want to compile for, not which OS you have. Indeed, you can perfectly compile and run a 32-bit program on a 64-bit Windows. So you'll most likely want to target 32-bit platforms, to have the largest possible audience. Choose 64-bit packages only if you have good reasons. **The compiler versions have to match 100%!** Here are links to the specific MinGW compiler versions used to build the provided packages: TDM 4.9.2 (32-bit), TDM 4.9.2 (64-bit), MinGW Builds 6.1.0 (32-bit), MinGW Builds 6.1.0 (64-bit)			
Linux	GCC - 64-bit	Download | 11.7 MB		
	On Linux, if you have a 64-bit OS then you have the 64-bit toolchain installed by default. Compiling for 32-bit is possible, but you have to install specific packages and/or use specific compiler options to do so. So downloading the 64-bit libraries is the easiest solution if you're on a 64-bit Linux. If you require a 32-bit build of SFML you'll have to build it yourself.			
Mac OS X	Clang - 64-bit OS X 10.7+, compatible with C++11 and libc++)	Download | 5.34 MB		
	Mac OS X libraries are only compatible with 64-bit systems.			
All	Source code	Download | 23.8 MB		
	HTML Documentation	Download | 1.48 MB		

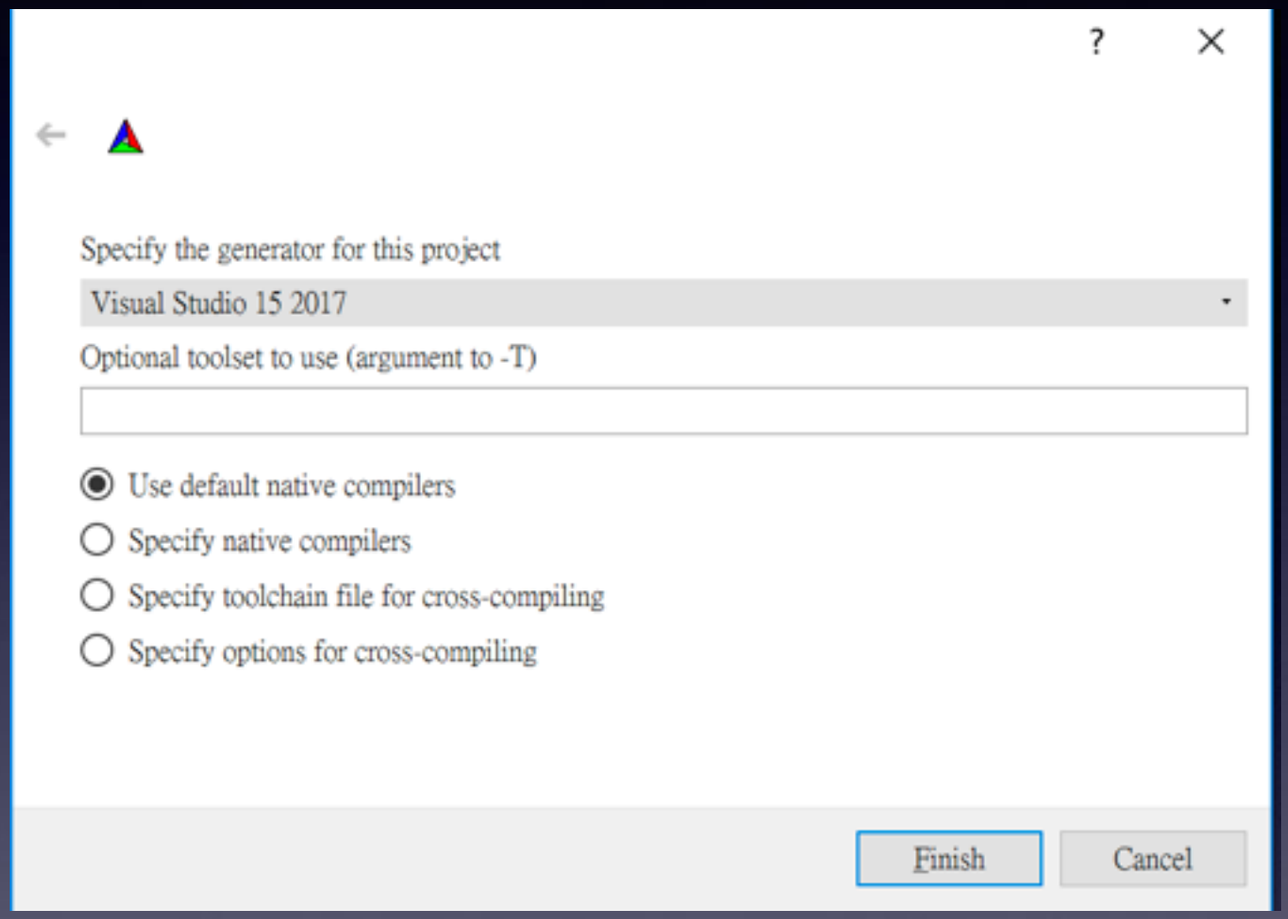
- 2017沒有直接支援，請下載Source Code
- 與此同時請安裝好Cmake https://cmake.org/

CMake in SFML 2



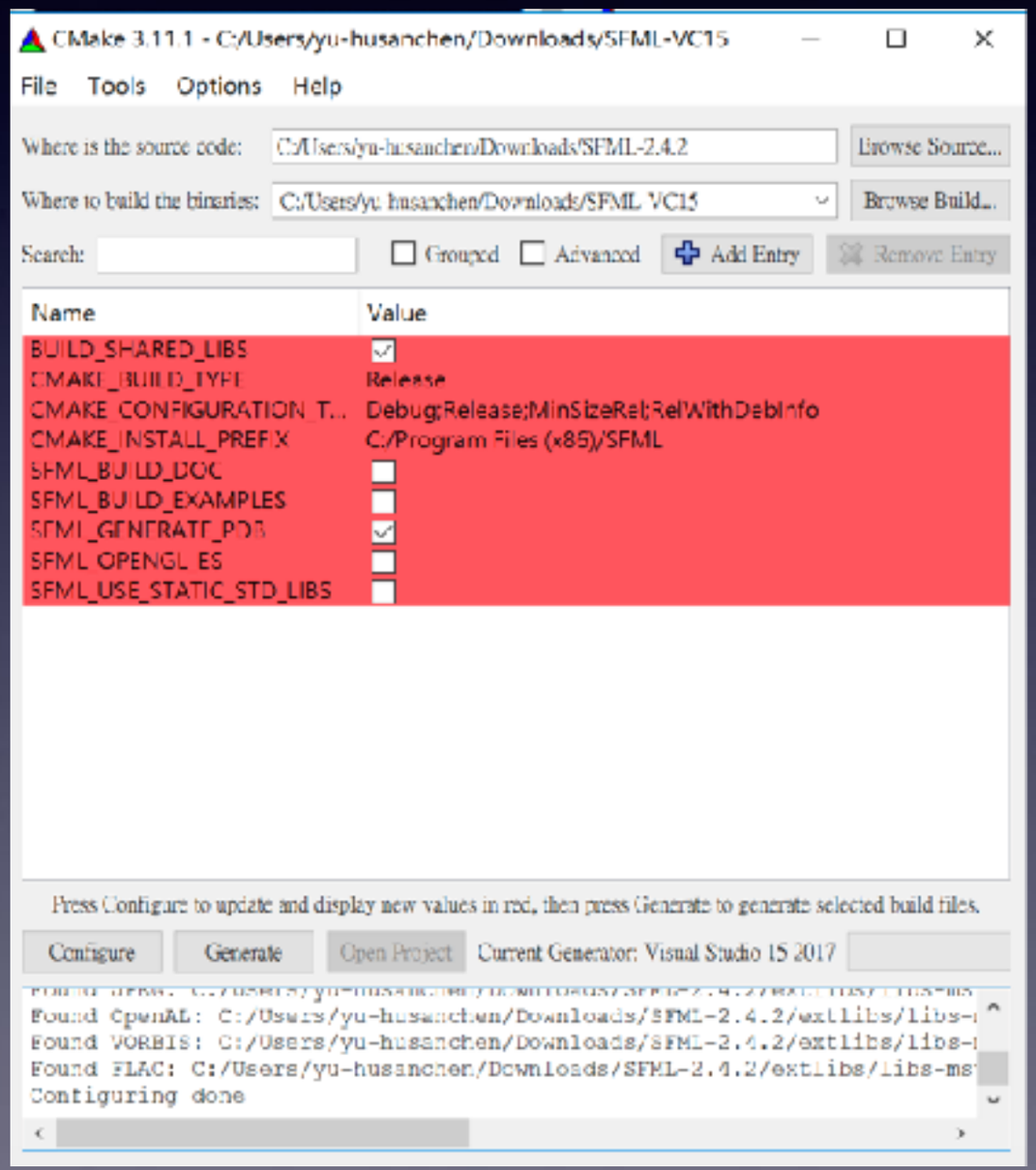
- 選擇source code 路徑與輸出路徑
  記得用資料夾裝起來（不然會很多檔案直接炸開）
  點選下面的Configure繼續

CMake in SFML 3



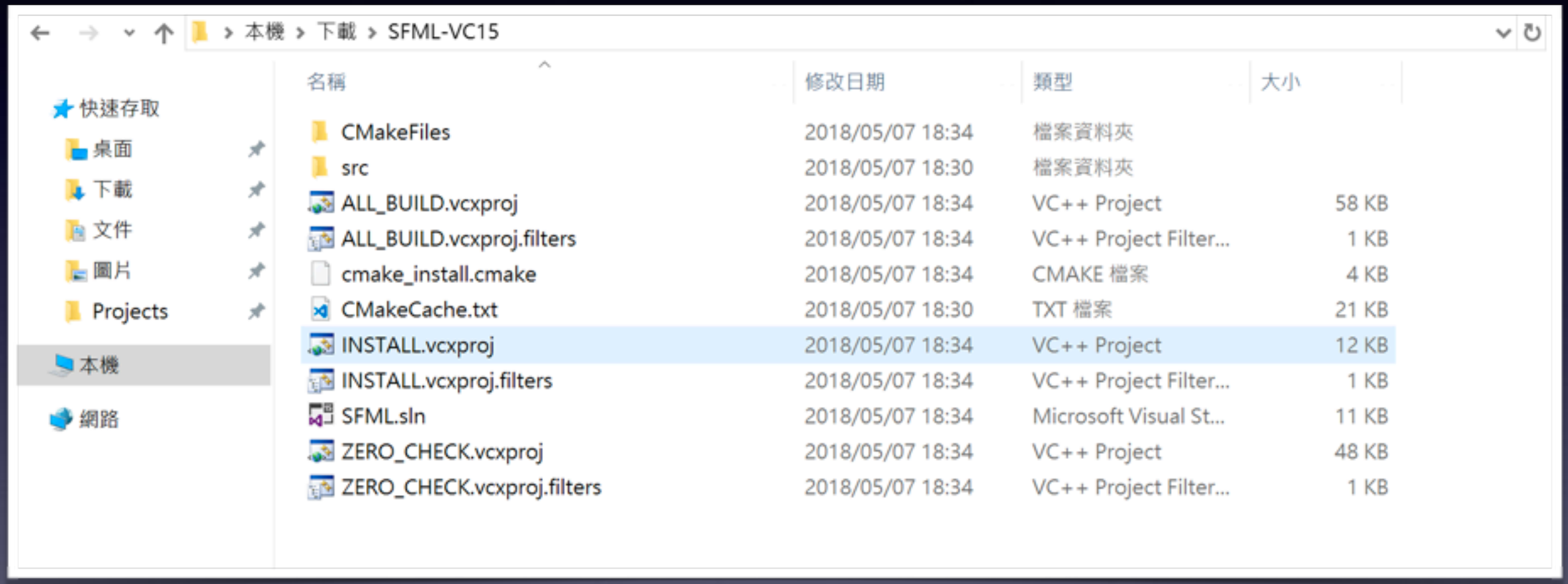
- 確定版本是Visual Studio 15 2017或是Visual Studio 2017 Win64，點Finish
  選定一個平台後後面建置也必須選擇相同的環境

CMake in SFML 4



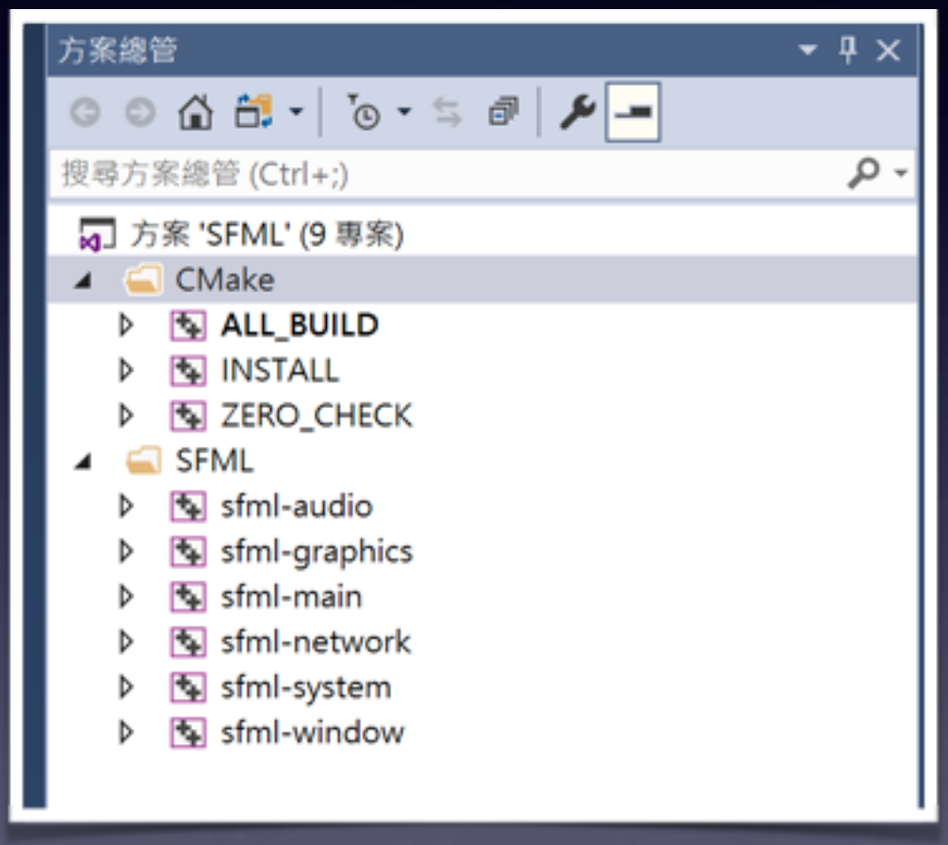
- 看到一堆紅色不要緊張，下面已經顯示Configuring done表示一切就緒
- 不需要更動任何東西，點Generate

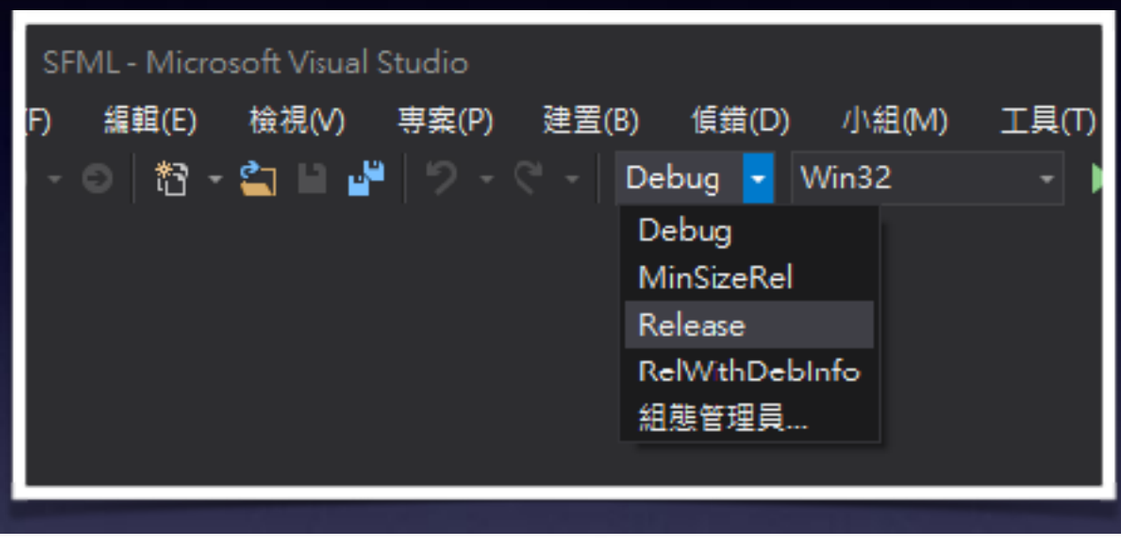
CMake in SFML 5

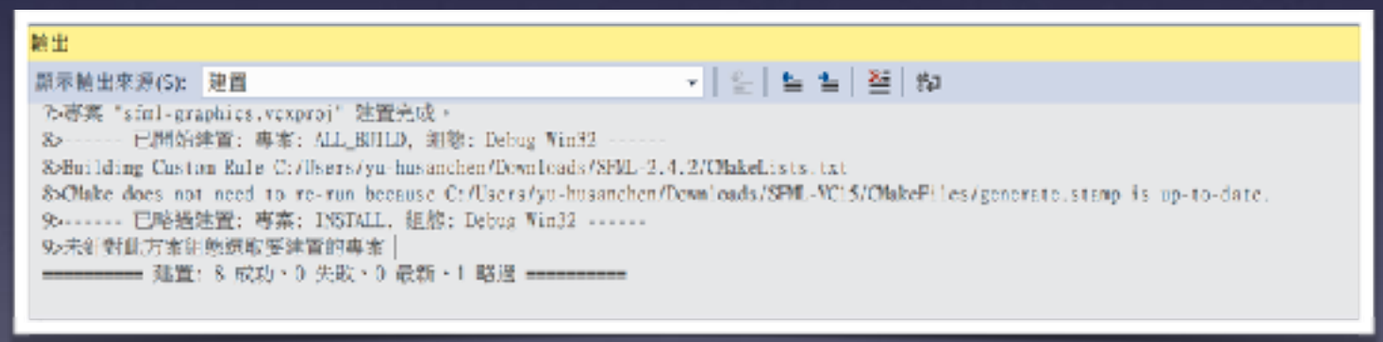


- 完成後回到剛剛的目標資料夾，雙擊sln檔開啟專案，進行後續操作

CMake in SFML 6

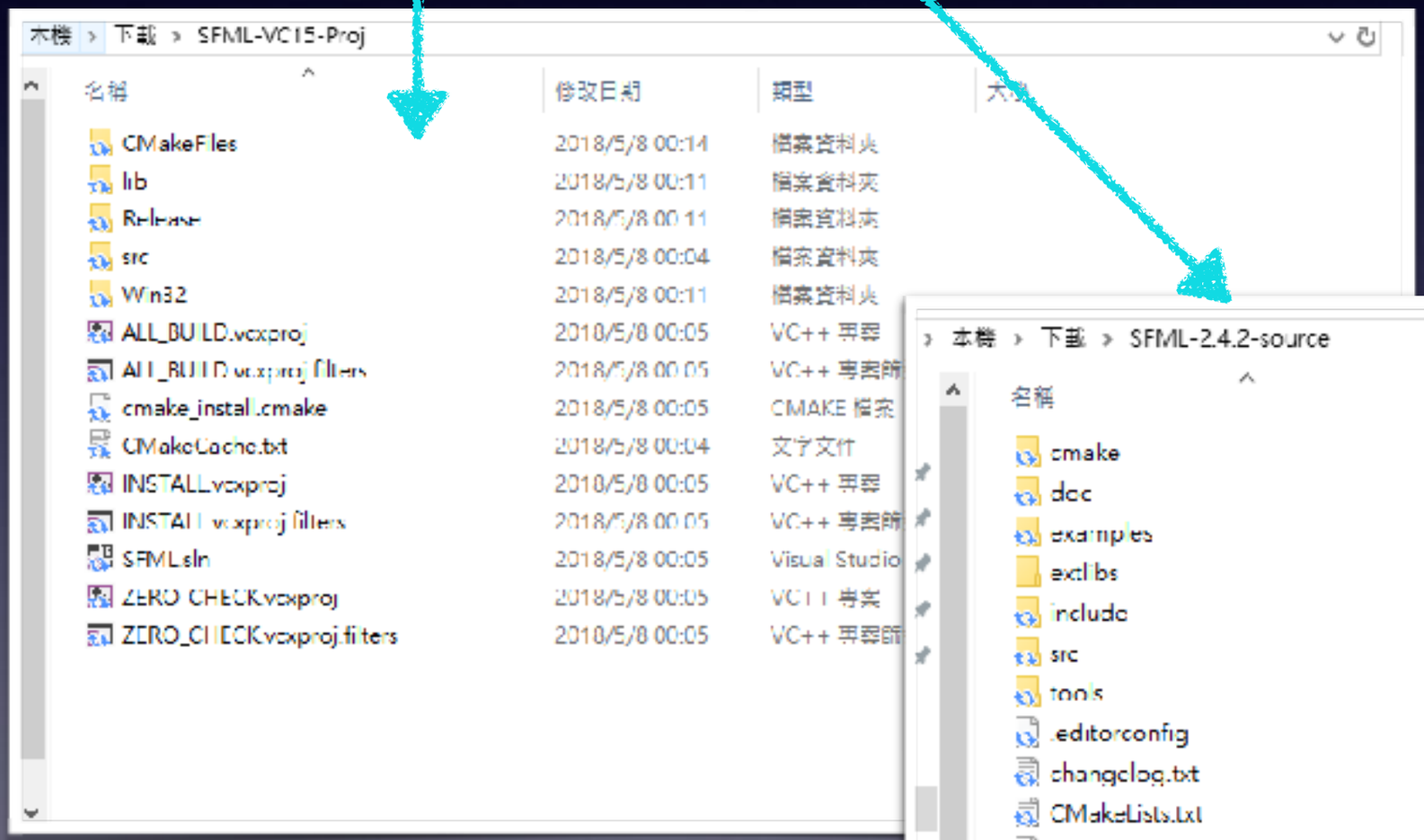


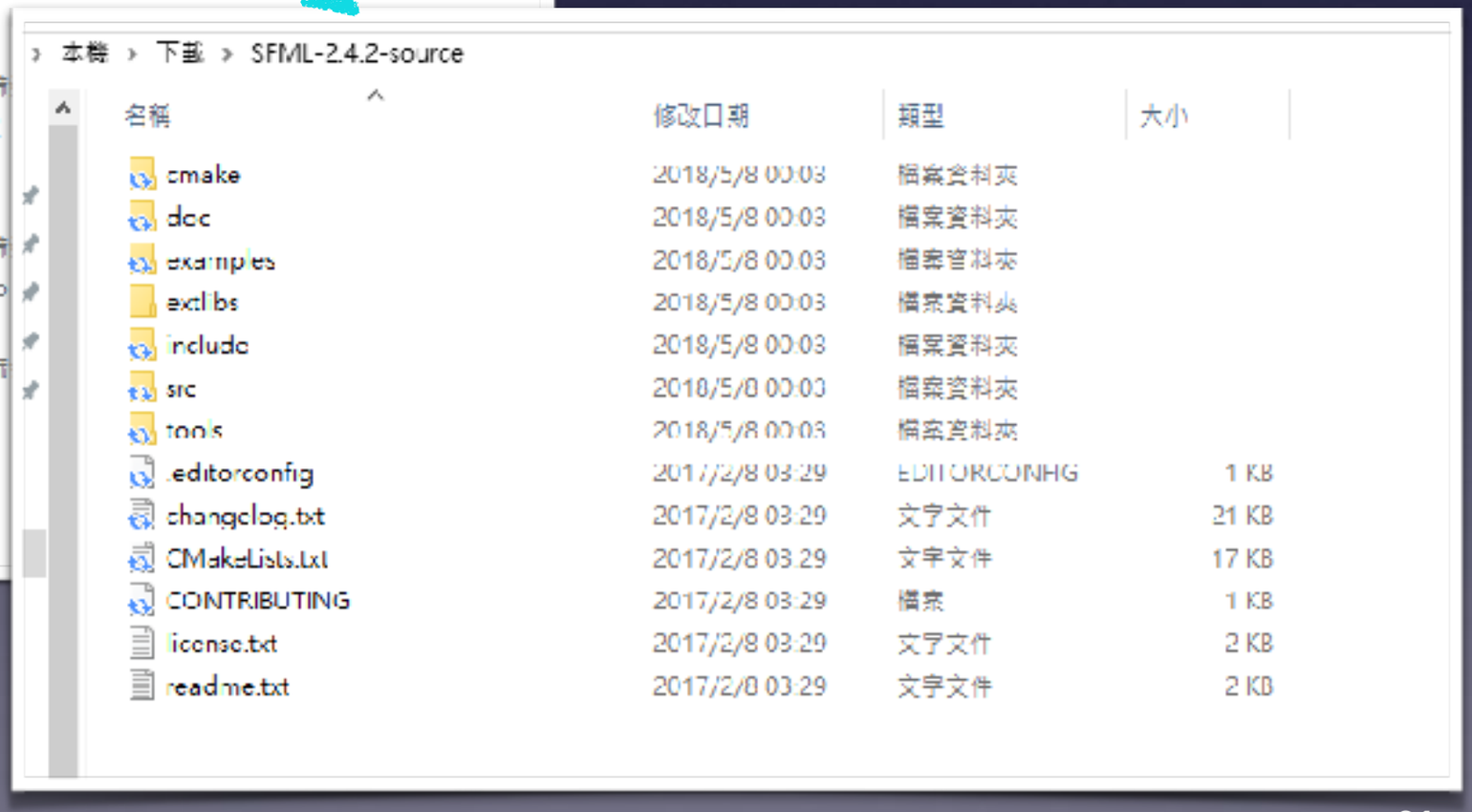




- 分別以Debug及Release作為目標平台，建置專案(Ctrl + Shift + B)
  缺少任何一個都會使後續寫SFML程式時產生LNK2019 Error
  （留意目標環境是Win32 or x64）

- 完成之後現在你有兩個東西
  原來的source code以及剛剛用Visual Studio 編譯過的專案檔

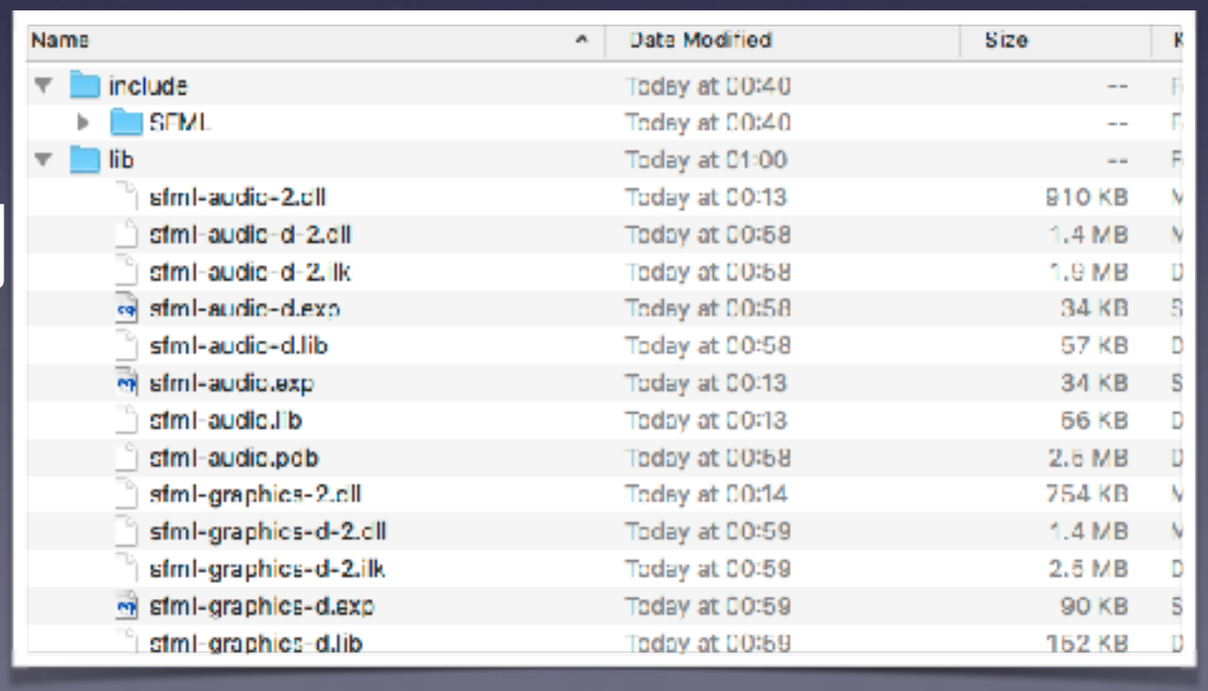




CMake in SFML 7

- 建立一個**新的資料夾**
  從source code複製include資料夾
  從**專案資料夾**中複製lib\Debug及lib\Release的所有資料到**新的資料夾**的lib目錄下

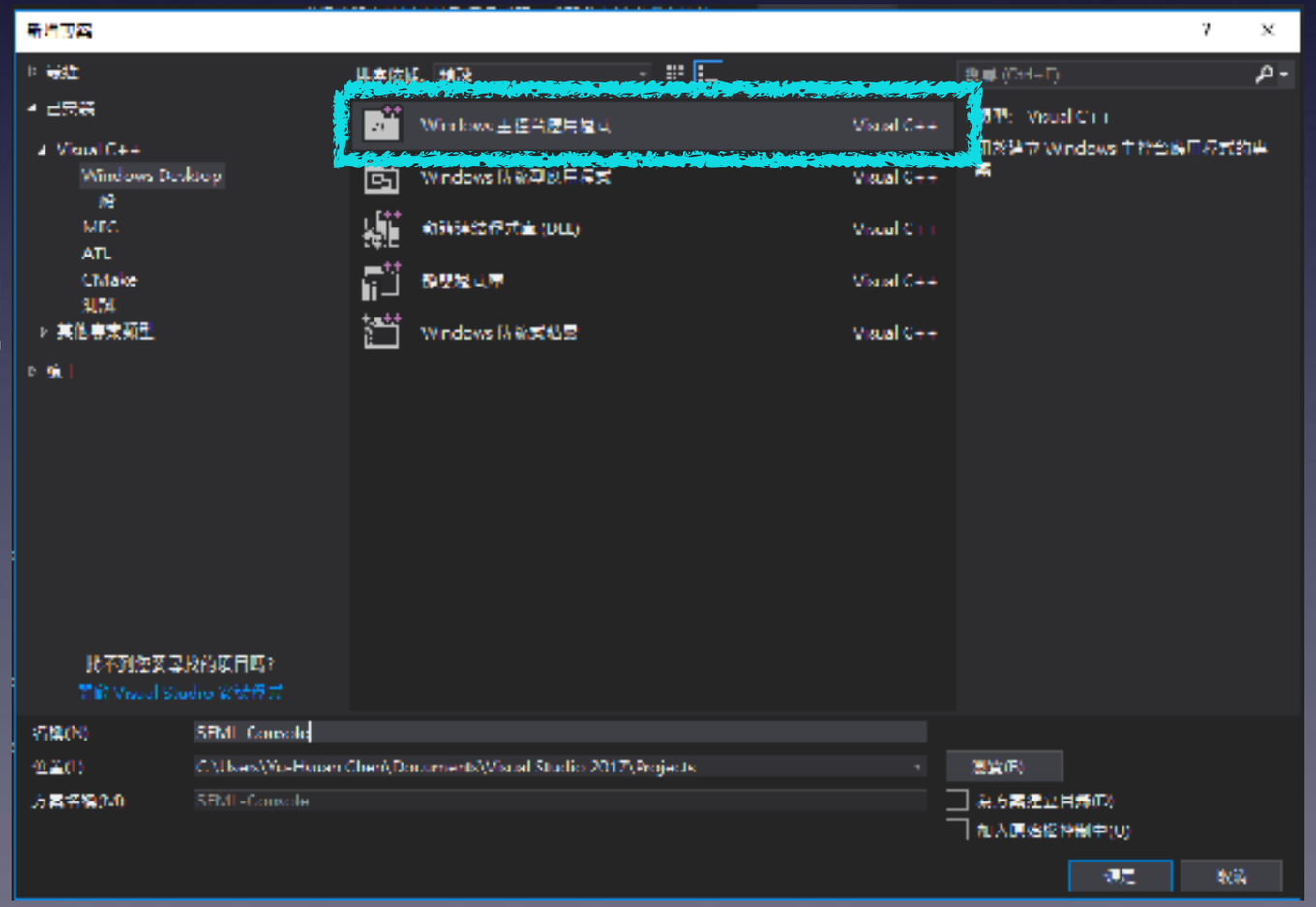
- 結構如右圖
  完成後這個資料夾就是我們自己製作的SFML Library
  一樣將它放到C:\底下



CMake in SFML 8

- 與2010需要做的事情相同，必須要將剛剛的資料夾加入系統環境變數
  請加入：C:\SFML-2.4.2\lib

- 開啟Visual Studio 2017 建立一個Windows 主控台應用程式 的專案



CMake in SFML 9

- 專案>屬性導入相關標頭檔
- 組態屬性>C/C++>一般: 其他Include目錄
  （組態：所有組態）
  加入 C:\SFML-2.4.2\include
- 組態屬性>連結器>一般: 其他程式庫目錄
  （組態：所有組態）
  加入 C:\SFML-2.4.2\lib

- 組態屬性>連結器>輸入: 其他相依性 （組態：Debug）
  加入
  sfml-graphics-d.lib; sfml-window-d.lib; sfml-system-d.lib; sfml-audio-d.lib; sfml-network-d.lib; kernel32.lib; user32.lib; gdi32.lib; winspool.lib; comdlg32.lib; advapi32.lib; shell32.lib; ole32.lib; oleaut32.lib; uuid.lib; odbc32.lib; odbccp32.lib; %(AdditionalDependencies)

- 組態屬性>連結器>輸入: 其他相依性 （組態：Release）
  加入
  sfml-graphics.lib;sfml-window.lib; sfml-system.lib; sfml-audio.lib; sfml-network.lib;kernel32.lib; user32.lib; gdi32.lib; winspool.lib; comdlg32.lib; advapi32.lib;shell32.lib; ole32.lib; oleaut32.lib; uuid.lib; odbc32.lib; odbccp32.lib;%(AdditionalDependencies)

- 留意不要把原來的東西給複寫了

做到這裡，與前面的2010操作一樣

你的測試程式應該可以順利編譯執行了