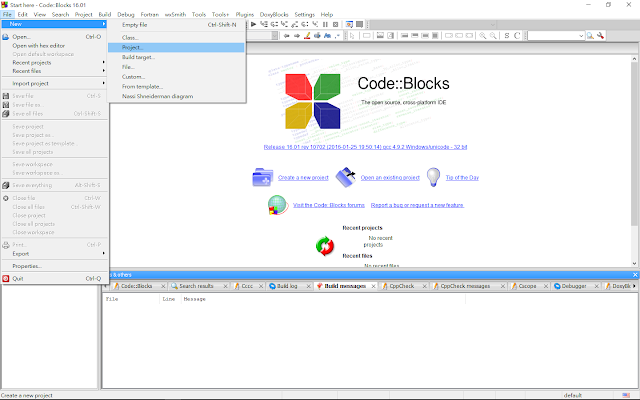
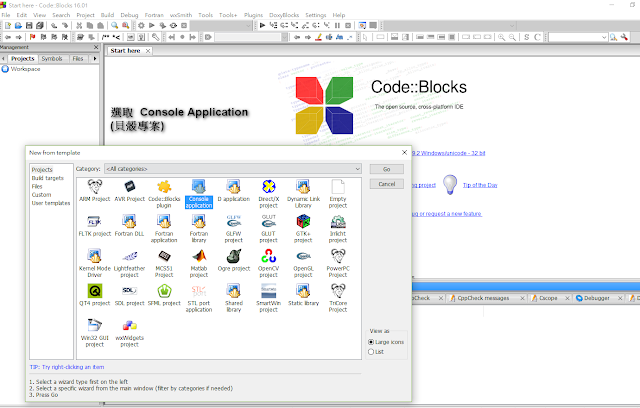
Cmake 主要是常用的跨平台編譯器  
<https://coolmandiary.blogspot.com/2017/11/cmakeclibrarydlllib.html>

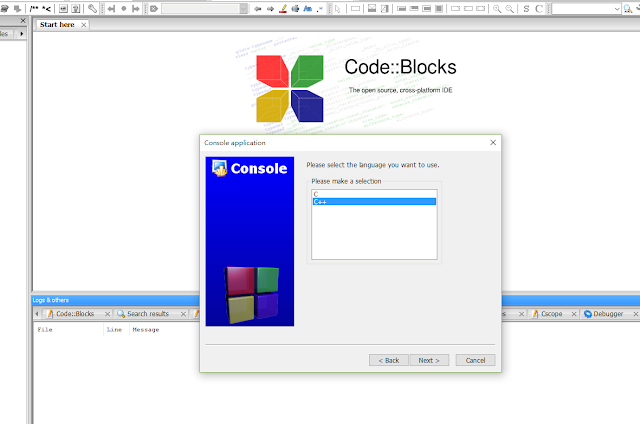
以下我們就用CodeBlock寫一支簡單HelloWorld輸出的小程式  
再次溫故悉知一次寫好的程式碼到封裝至運行的整體流程喔!!!!  
  
File --> New Project

[](https://4.bp.blogspot.com/-k-KqI9aKPH8/WgZRXrEouII/AAAAAAAAK_Q/L-wItr2mbMA7Sy54vM6jBc3fGJ9R5DnmgCLcBGAs/s1600/a1.png)

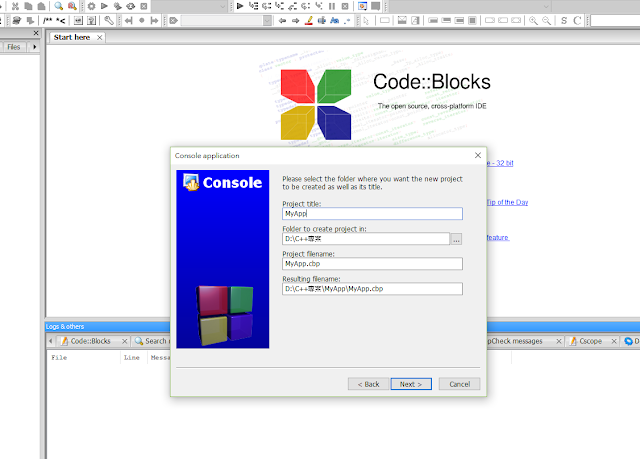
 選擇 Console Application

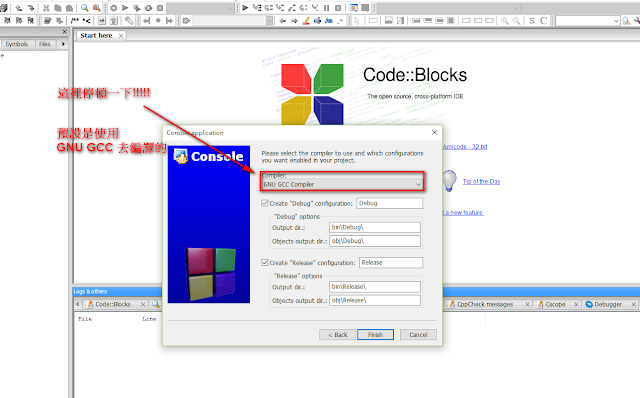
[](https://2.bp.blogspot.com/-R9UlsfIUykQ/WgZRYlLP5RI/AAAAAAAAK_U/5Zg7TzgGxis9FDtqGKprhLIAcJA6UTaFACLcBGAs/s1600/e1.png)

 在此選擇C++

[](https://4.bp.blogspot.com/-Lx3xTR8Pb9g/WgZRZ0zq0_I/AAAAAAAAK_Y/_BRZLhlRkwEDiypey_jx8zbVPnOzj8PYgCLcBGAs/s1600/a2.png)

 將專案命名好

[](https://2.bp.blogspot.com/-TTs3N5b-qJQ/WgZRa2Y10LI/AAAAAAAAK_c/wbDWTp1oyqYYgLB-fSnZMkBYA8PBJ0q0ACLcBGAs/s1600/gw.png)

[](https://4.bp.blogspot.com/-qJKu2E3tDw4/WgZSAiARB1I/AAAAAAAAK_k/P5wa2-F2d8Aq5PEvroU3chXMO_VpxdDWACLcBGAs/s1600/b.png)

這裡的 GNU GCC Compiler  
指的就是一種編譯器(講白話一些就是翻譯官  翻譯人員)  
**GCC**主要全稱就是  **G**NU **C**ompiler **C**ollection  
**GNU編譯器套裝**  
是以GPL及LGPL授權條款所發行的自由軟體  
GCC (以C語言編譯器為主要發展特色/1985年由理察·馬修·斯托曼開始發展)  
也常被認為是**跨平台編譯器的一種標準**。  
  
  
  
GNU(商標主要是一隻牛羚)  ---> 主要是一種自由作業系統，其內容軟體  
完全以**GPL(General Public License)**方式釋出  
也就是所謂的  「通用授權條款」  
GPL授予程式接受人以下自由權利:  
  \* 以任何目的運行此程式的自由；  
  \* 再複製的自由；  
  \* 改進此程式，並公開發布改進的自由（前提是能得到源代碼）。

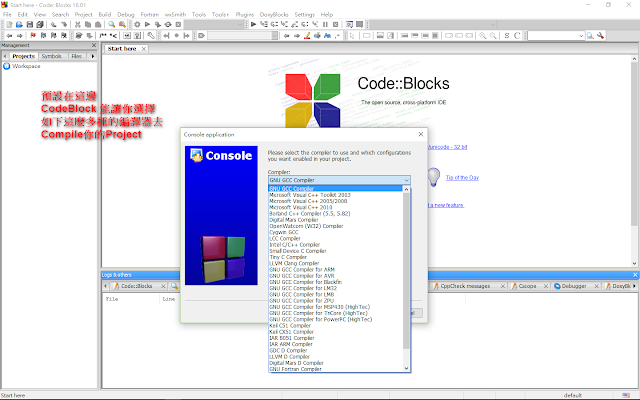
**==>**尋求確保上述自由能在**複製軟體**及**衍生作品**中得到保障

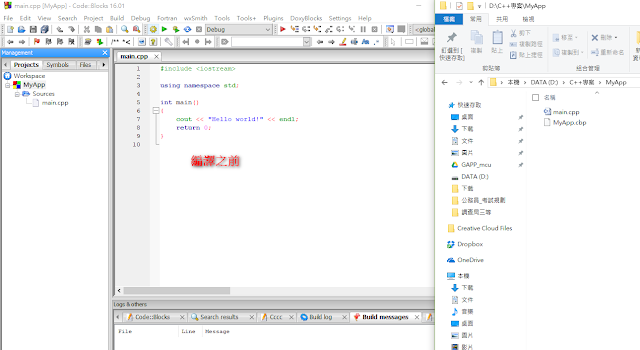
也是一般廣泛大眾使用的很常聽到的**免費軟體授權條款**

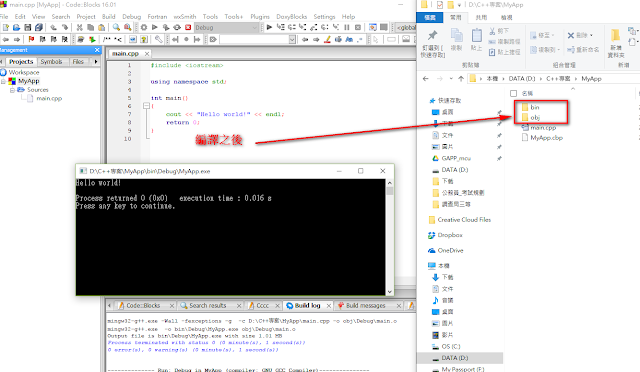
[一張含有 草, 戶外, 哺乳動物, 領域 的圖片

自動產生的描述](https://4.bp.blogspot.com/-UDePMB9LAmY/WgZdxjo5A0I/AAAAAAAALAg/LjbP4oixEXot-ve3zyQg4-LhOPbsgafVgCLcBGAs/s1600/Wildebeest_Connochaetes_taurinus_in_Tanzania_2793_Nevit.jpg)[一張含有 文字 的圖片

自動產生的描述](https://4.bp.blogspot.com/-noPLbmbgFSI/WgZap7XUQ1I/AAAAAAAALAU/Fhnz_bhZ2xs4qwUldgW-a5zsHhCquNjqQCLcBGAs/s1600/535px-Heckert_GNU_white.svg.png)

[](https://2.bp.blogspot.com/-oXlbGuPamIM/WgZSUAaheaI/AAAAAAAAK_o/Ges_1OPXmDg-6mnEThfSfKEDrG62ai1iACLcBGAs/s1600/va.png)

[](https://4.bp.blogspot.com/--6VddkdrQe8/WgZXpMFDdeI/AAAAAAAAK_4/4M0MaJillTART45DMfByq2y7S1_Wn0s4gCLcBGAs/s1600/vf.png)

[](https://4.bp.blogspot.com/-M3SsnqeZfwk/WgZXqM8JhFI/AAAAAAAAK_8/YGOmM49RdBokGYMle3vabYfvplXIJuulQCLcBGAs/s1600/f.png)

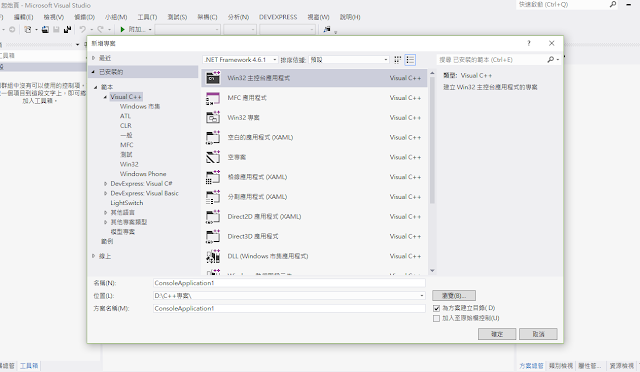
就多出了  bin 及 obj目錄

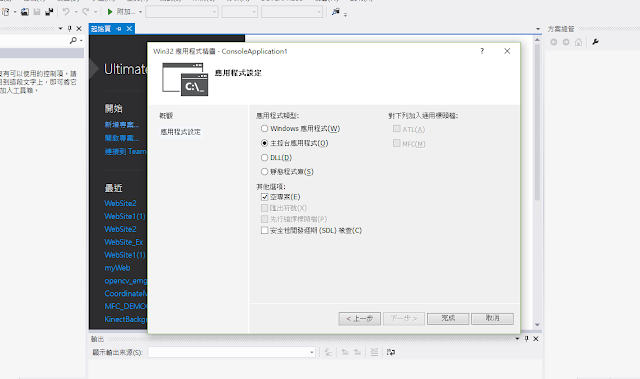
[一張含有 文字 的圖片

自動產生的描述](https://3.bp.blogspot.com/-zyYyMqgkDU0/WgZYNiH-UBI/AAAAAAAALAI/VLIZcCjKL1I4-QlTWQ7R0VKmnB7KcLZtACLcBGAs/s1600/e22.png)

bin目錄主要放置 經過Compiler翻譯成電腦看的懂的機器碼(執行檔  .exe)  
也就是二進制binrary的英文縮寫，主要用來存放編譯的結果  
最初C編譯的程序檔案都是二進制的File  
  
  
若你使用visual studio編譯則會分成有Debug(測試區)和Release(正是區)兩個版本  
分別對應的檔案夾為bin/Debug和bin/Release  
  
[一張含有 文字 的圖片

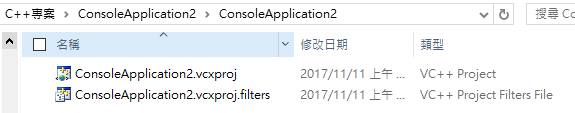
自動產生的描述](https://3.bp.blogspot.com/-GpNWkHq5OVA/WgZYM89yEtI/AAAAAAAALAE/7WJjENkLpyctK5vgK2jaJEdebmyjNPQhQCLcBGAs/s1600/e11.png)  
  
obj目錄則是放了   .o檔案  
  
那基本上你可能會好奇這兩個目錄的差異  
還有打開基本上都是像加密過的亂碼一樣看不太懂  
  
  
以下在來看一個  Visual Studio 在編譯時過程

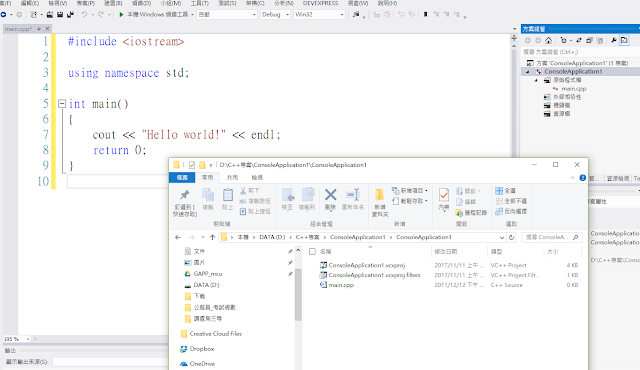
[](https://1.bp.blogspot.com/-lwheta4YhLk/WgZjGLxL_SI/AAAAAAAALA0/qsQhDKr-aJg6ioN0zXhTsE7krhAJTbxFACLcBGAs/s1600/k1.png)

[](https://1.bp.blogspot.com/-M9-v45wEP9w/WgZjGwAbD4I/AAAAAAAALA4/ETdQAtmiqyssZbvAz-OPLjfLZRHtqHvIACLcBGAs/s1600/k2.png)

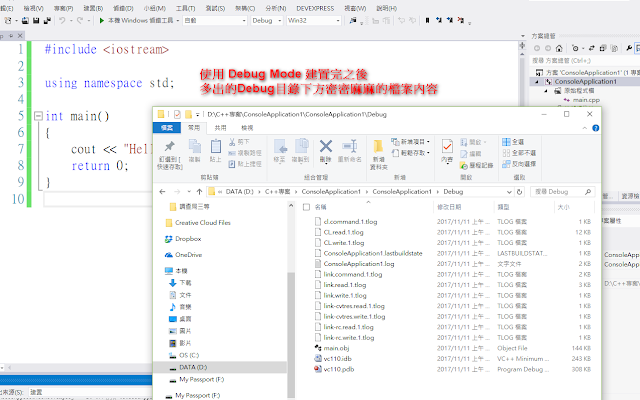
預設建置好一個新的C++ Win32 Console方案中底下的專案時  
只會有  
[一張含有 文字 的圖片

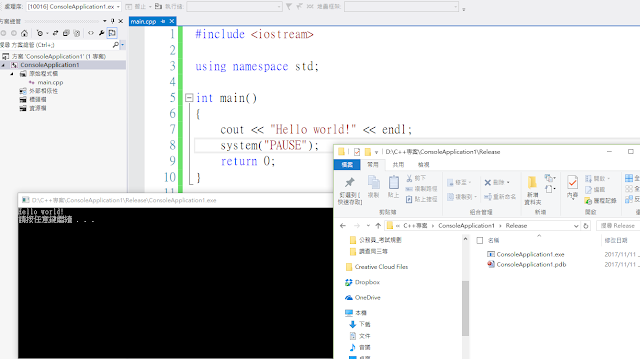
自動產生的描述](https://4.bp.blogspot.com/-T_BVCNqFq_E/WgZksBfpxII/AAAAAAAALBQ/3Y2fd4RShDwgPk5Z6iFXOlZi2IBwNt9IQCLcBGAs/s1600/gg.png)  
一個ConsoleApplication方案 的Folder  
底下分  
三個檔  
ConsoleApplication專案名稱.opensdf  
ConsoleApplication專案名稱.sln  
ConsoleApplication專案名稱.**v11**.suo   ---->因為是使用**vs2012** 所以對應的是v11喔!!!!  
一個目錄  
ConsoleApplication專案名稱的Folder  
(和專案同名的子目錄)

[](https://3.bp.blogspot.com/-OyQU72ewuvE/WgZlqEQX24I/AAAAAAAALBc/Dc69vu-cANM3BCDjPwyZwVpxiNJhbBjbwCLcBGAs/s1600/ggg.png)

[](https://1.bp.blogspot.com/-3vSpEGG62Ig/WgZjHm6HwEI/AAAAAAAALA8/InnF0EtWgw8fyV4t6r50cI34nhUjyPCaQCLcBGAs/s1600/k3.png)

 用DebugMode建置後  
就在此層多出Debug字眼目錄

[](https://1.bp.blogspot.com/-ARlL0A5DBjU/WgZjIot4vXI/AAAAAAAALBA/bwCZDDApUPE8FFabOEbgn1fnXxvjqe0FwCLcBGAs/s1600/hg1.png)

[](https://2.bp.blogspot.com/-Ve_ew-xU2kk/WgZjO0Z3qwI/AAAAAAAALBE/5ZSljR7yvtkwRROh9IgF4o80ZXikvIq8ACLcBGAs/s1600/bbb.png)

====================================================================  
如下展示如何搭配  git  , Cmake  
去做開放庫gFlag的建置(Build)  
  
工具列表  
1.Cmake  
<https://cmake.org/download/>  
2.git  
[https://git-scm.com](https://git-scm.com/)  
或  小烏龜  
[https://tortoisegit.org](https://tortoisegit.org/)  
3.gFlags Library  
The gflags package contains a C++ library that implements commandline flags processing.  
<https://github.com/gflags/gflags>  
  
4.visual studio (當前示範使用2012)  
  
  
  
  
====================================================================  
**gFlags Library**  
  
Step1.

[一張含有 文字 的圖片

自動產生的描述](https://4.bp.blogspot.com/-FNQKd72NrkI/WgZwOFnCTfI/AAAAAAAALBs/-_W6hquLg9MgWgpGOpef7k-FpiieeXreACLcBGAs/s1600/e11ss.png)

Step2.開啟  git bash window  
指令:  git clone 剛剛複製的URL

[一張含有 文字, 水, 螢幕擷取畫面, 岸 的圖片

自動產生的描述](https://3.bp.blogspot.com/-npMeoxygEbM/WgZxfxFpPPI/AAAAAAAALB0/FXlFcS433D4_ymKUMWnmLZTOh-AyfcxEwCLcBGAs/s1600/tttfff.png)

或用  小烏龜  也可以

[一張含有 文字 的圖片

自動產生的描述](https://4.bp.blogspot.com/-zcX00x40POY/WgZx4u7m2wI/AAAAAAAALB4/PqhtEse_lygvAXTtOxrQlzbFBjioFOEEQCLcBGAs/s1600/nh.png)

[一張含有 文字 的圖片

自動產生的描述](https://1.bp.blogspot.com/-5colDG5sYLQ/WgZx5Cc-6JI/AAAAAAAALB8/8iHmTxEOnRcEbQ7LHsxe-_VJmu6MUZD1QCLcBGAs/s1600/nhbb.png)

[一張含有 文字 的圖片

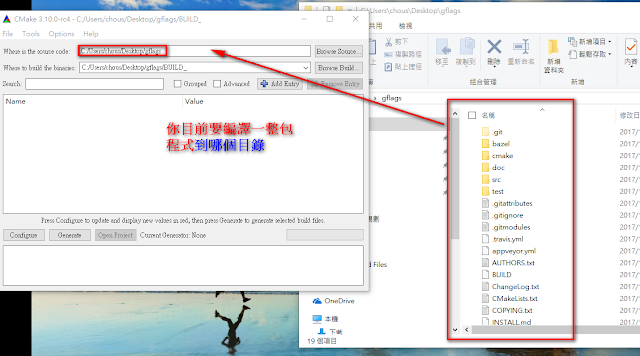
自動產生的描述](https://4.bp.blogspot.com/-5colDG5sYLQ/WgZx5Cc-6JI/AAAAAAAALB8/MOUSiMwKJ0AFDSEHi2UqAWbIig5ZDWV5wCEwYBhgL/s1600/nhbb.png)

**Cmake下載及安裝流程**  
Step1.進官網連結下載  
Cmake  
<https://cmake.org/download/>

[一張含有 文字 的圖片

自動產生的描述](https://1.bp.blogspot.com/-vtBbj2dWaLg/WgCPmjvD8vI/AAAAAAAAK-w/8LOzxrR-L34zLHKg33rtFAC1OK9NoCqBwCLcBGAs/s1600/d1.png)

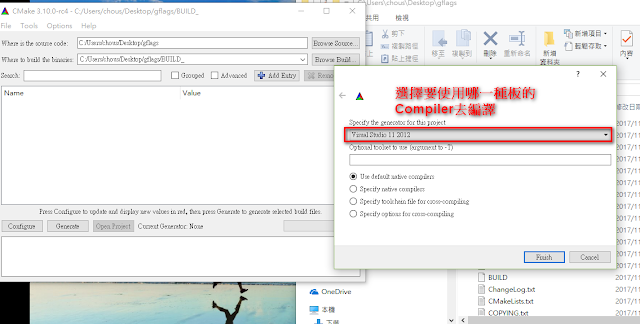
Step2.使用Cmake編譯一支程式  
開啟 Cmake Gui

[](https://2.bp.blogspot.com/-2WCZMM17e8A/WgZy9Niq6nI/AAAAAAAALCI/RF7mjt7AFboOfYuNWYajLVcczrQf8lr8ACLcBGAs/s1600/vv.png)

琣組態選好的來源及存放目的目錄路徑後  
按下  Configure

[一張含有 文字 的圖片

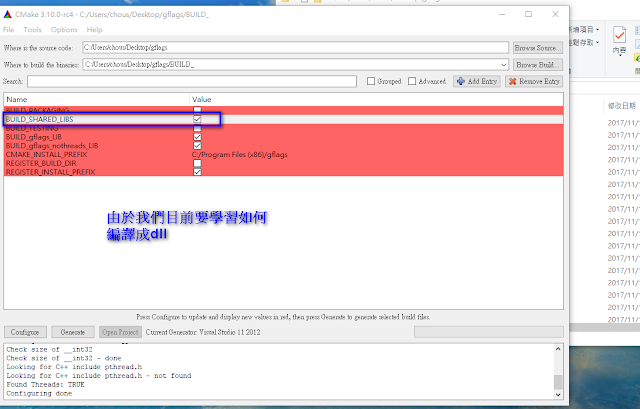
自動產生的描述](https://4.bp.blogspot.com/-D13PFqMsBew/WgZzQ62DiHI/AAAAAAAALCM/3xrvNLaVm1AWjWG24bFHMarPCgPHPQIgACLcBGAs/s1600/nn.png)

[](https://3.bp.blogspot.com/-ITEODmw3ASI/WgZz3V6jASI/AAAAAAAALCU/z5c_ZezfBHw5SdxPo2bn0C9Q4NiMhs7fACLcBGAs/s1600/gfa.png)

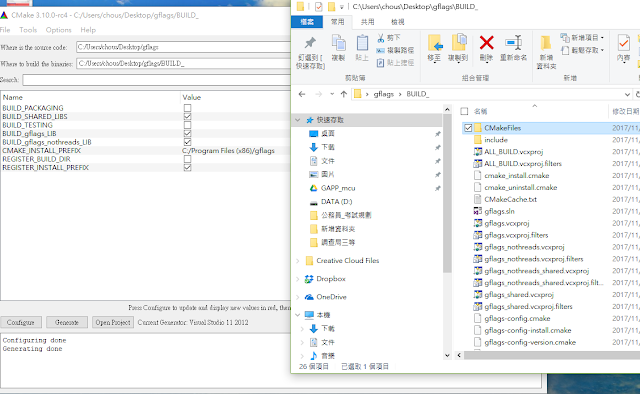
[一張含有 文字 的圖片

自動產生的描述](https://1.bp.blogspot.com/-_fkqTw6vqeg/WgZz4P7nUTI/AAAAAAAALCY/G2T1Lk_fJT0NjolhNlVvDwAApkAWGcQOgCLcBGAs/s1600/vva.png)

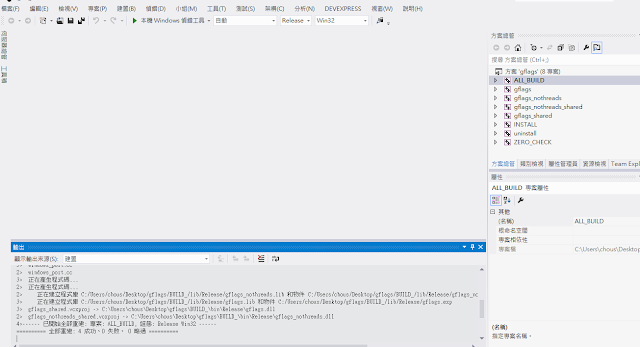
當你按完  Configure之後  
會生成紅色背景一系列的選項  
**BUILD\_SHARED\_LIBS**  
**------> 這裡打勾是for 動態連結庫(DLL)**  
若沒有要特別生成出動態  -----> 照預設選項勾選擇是 FOR 靜態的(.LIB)

[](https://2.bp.blogspot.com/-9fgDLrJRdLg/WgZ0Ur8xG6I/AAAAAAAALCc/sleFKdjF8sg8dMVV8HXJPnAxLNlwS_9wACLcBGAs/s1600/c.png)

選擇 .dll  
可達到  Smaller Executable 的特性  
當選完按下 Generate之後  
就會生出一堆 vcxproj的相關檔案

[](https://2.bp.blogspot.com/-xL57qmIyZII/WgZ3Sq8UiFI/AAAAAAAALCo/HUkmSA-E5mcrqRZJRxAZPtPpkN2y5v4QwCLcBGAs/s1600/m.png)

這裡我們要打開  sln檔案  
由於我剛才選的是用vs2012去組態  
所以若你使用  vs2013 ,2015去開  
就會告知你要不要升級的訊息喔!!!!!這點切記!!!!!  
要使用對應版去開或是Update它

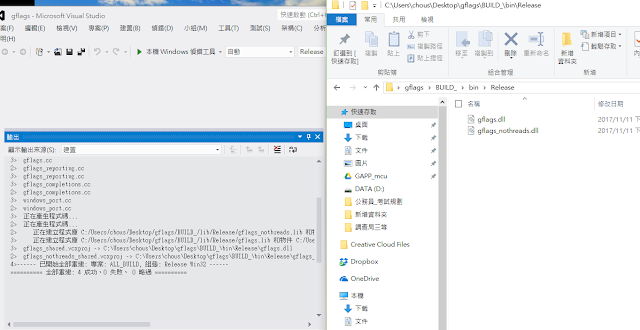
[](https://3.bp.blogspot.com/-fxK7qx9r15A/WgZ4ZhbNbSI/AAAAAAAALCw/xcxuww9vINwoQ6vIekI6wFESR6lXkhx6wCLcBGAs/s1600/jyjy.png)

這裡在 Build好之後你會看到  
多出來的  Release目錄 及 Win32目錄字眼 跟其他目錄  
就是因為我們剛才在vs2012選擇 用 正式區(Release) 及 Win32 Mode去編譯關係

[一張含有 文字 的圖片

自動產生的描述](https://2.bp.blogspot.com/-aSvY4Qg73r8/WgZ4zBQi2OI/AAAAAAAALC4/afPwLmivyYIpOvo5SCIDgspOhQkMsv69QCLcBGAs/s1600/nhb.png)

那基本上 Win32那裡都是生成 Log的編譯歷程紀錄  
我們到   ..\gflags\BUILD\_\bin\Release  
底下就能看到剛才我們所編譯生成的dll了優~~~

[](https://4.bp.blogspot.com/-uAPAMdvfpvM/WgZ50ZYmOsI/AAAAAAAALDA/ArWp0FSrOfoG3XJkm0JuBvGQoUvHUGZTACLcBGAs/s1600/u.png)

=============================================================  
在看下去之前可以先參考之前寫的兩篇文章會更加有概念  
C#自建類庫項目DLL\_做一個自己的dll  
http://coolmandiary.blogspot.tw/2016/12/cdlldll.html  
  
C#\_DLL反編譯功能\_EXE(Application)和DLL(Application Extension)的差別在哪???  
\_DLL的意義\_反編譯目的  
http://coolmandiary.blogspot.tw/2017/04/cdllexeapplicationdllapplication.html  
=============================================================  
介紹到這你可能會有個疑問到底  
  
捨麼是 Shared Library? (就是指 DLL)  
捨麼是 Linking?  
捨麼是 Static Library?  
**Shared Library和Static Library差異**  
從外觀特徵(副檔名)講起  
  
**Static Library**  
**By Windows 平台 : .lib (Library)**  
**By Linux : .a   (Archive)**  
 **Shared(Dynamic) Library**  
By Windows 平台 : .dll (Dynamic Link Library)  
By Linux : .so (Shared Object)

[一張含有 文字, 信 的圖片

自動產生的描述](https://3.bp.blogspot.com/-zTDE1oASJ1A/WgZ8elzxD5I/AAAAAAAALDM/LPSSzUbaUAExt2YQTU2ItNEPslMImPBrwCLcBGAs/s1600/s.png)

從運行上來比較  
最早也是最常見到的 DLL  
主要是包在於 WIndows API中的  
(1)Kernel32.dll  
當中又包含了  
管理內存、Process跟Thread等等各個函數  
(2)User32.dll  
包含用來執行用戶介面任務  
(3)GDI32.dll  
畫圖、文字文字顯示用途

[一張含有 圖表 的圖片

自動產生的描述](https://4.bp.blogspot.com/-BAb5_fXNWp4/WgaQzHHCl0I/AAAAAAAALDo/L5401WqbtAscdtY1gD1IWZ-r3giTQ9wcQCLcBGAs/s1600/hy.png)

用動態連結庫(DLL)好處  
就是它是參照該程式庫的 Address(記憶體中某一塊位址)  
所以只要Compile一次就可以也不花時間在每次都要重新建置  
1.可用多種程式語言做開發撰寫(Ex:用VB寫介面 ,調用其他語言完成業務邏輯的DLL )  
2.提供二次開發衍生功能  
3.可節省磁碟空間  
4.助於資源共享(Shared)  
5.助於實踐應用程式Local化  
6.簡化專案控制管理  
  
相反的  
  
靜態連結庫  
假如你寫一個程式當中有一個function  
然後你想要將其功能服務連結至其他檔案的時候就會有檔案大小超載變過大的問題

[一張含有 圖表 的圖片

自動產生的描述](https://2.bp.blogspot.com/-2DZh9-qf0cA/WgaP-wcTDgI/AAAAAAAALDc/nLVgju-XPMI0ArM7YGRmexZkmzgsxsILQCLcBGAs/s1600/bg.png)

同樣的功能程式  
在檔案1 調用了 200次  
在檔案2 調用了 100次  
當彼此共同用Linker去包裝過後就會產生較大的執行程式檔案

[一張含有 圖表 的圖片

自動產生的描述](https://3.bp.blogspot.com/-k0ibDlGIbiE/WgaQC1oUNwI/AAAAAAAALDg/ELeqE0wRM34buvgGVoBJlVW4iIon9jmqACLcBGAs/s1600/bbg.png)

特徵就是程式碼比較大一包而且是整串Code  
直接Call (Copy過去才能使用)  
而且組態上可能還會遇到版本問題或是水土不服  
比較缺乏共享性質  
且十分佔據資源空間