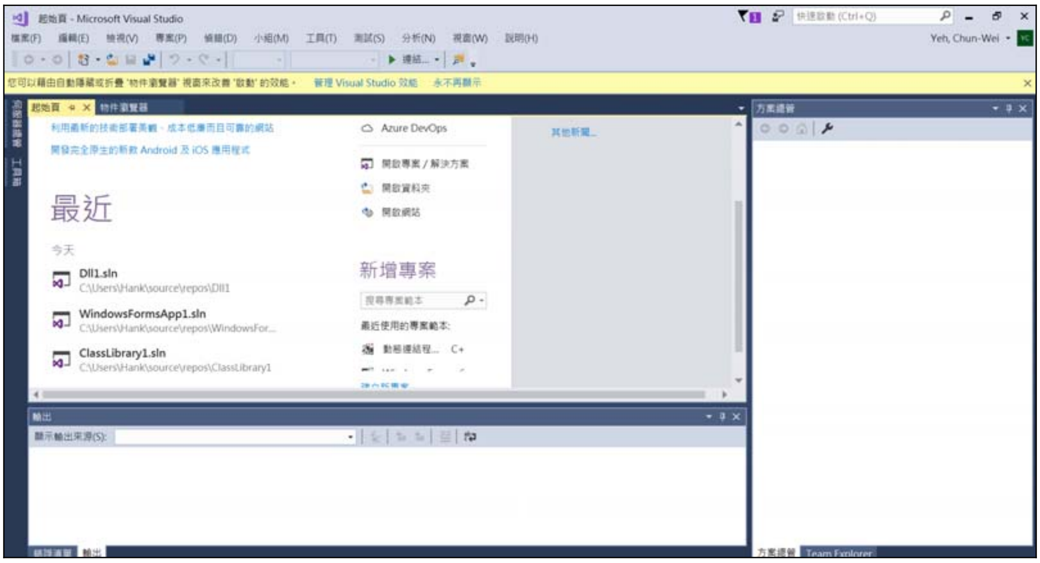
國立臺北科技大學自動化所 嵌入式工業機器視覺

Lab 4_Using P/Invoke for NImage

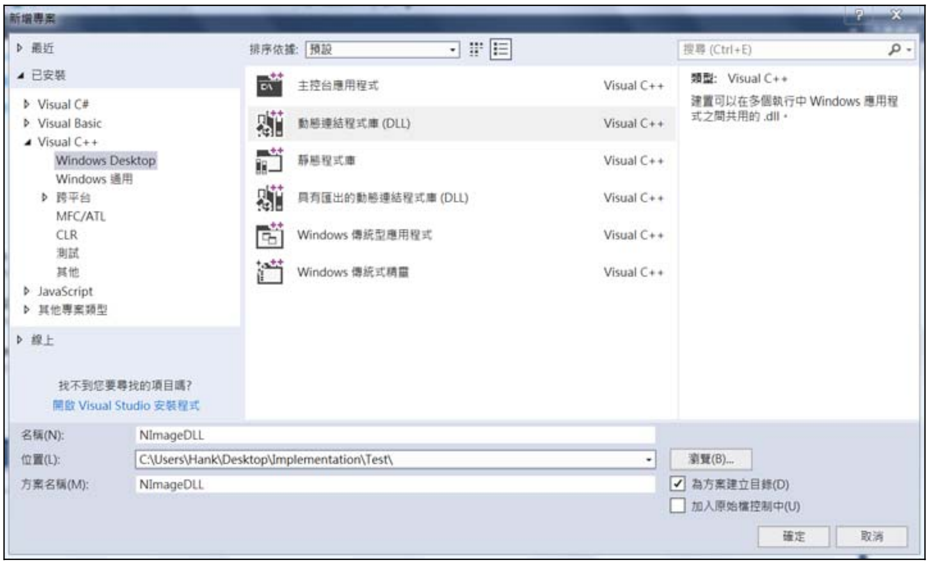
[Using P/Invoke for C++]

<u>The following steps describe how to create a dynamic library on Windows. These steps are for Microsoft Visual Studio 2017, although the steps are similar for other versions of Visual Studio.</u>

1. Select the menu File > New > Project



2. Select the Visual C++ -> Windows Desktop and the DLL option, and then put the project name and path.



類型: Visual C++

建置可以在多個執行中 Windows 應用程式之間共用的 .dll。

名稱(N): NImageDLL

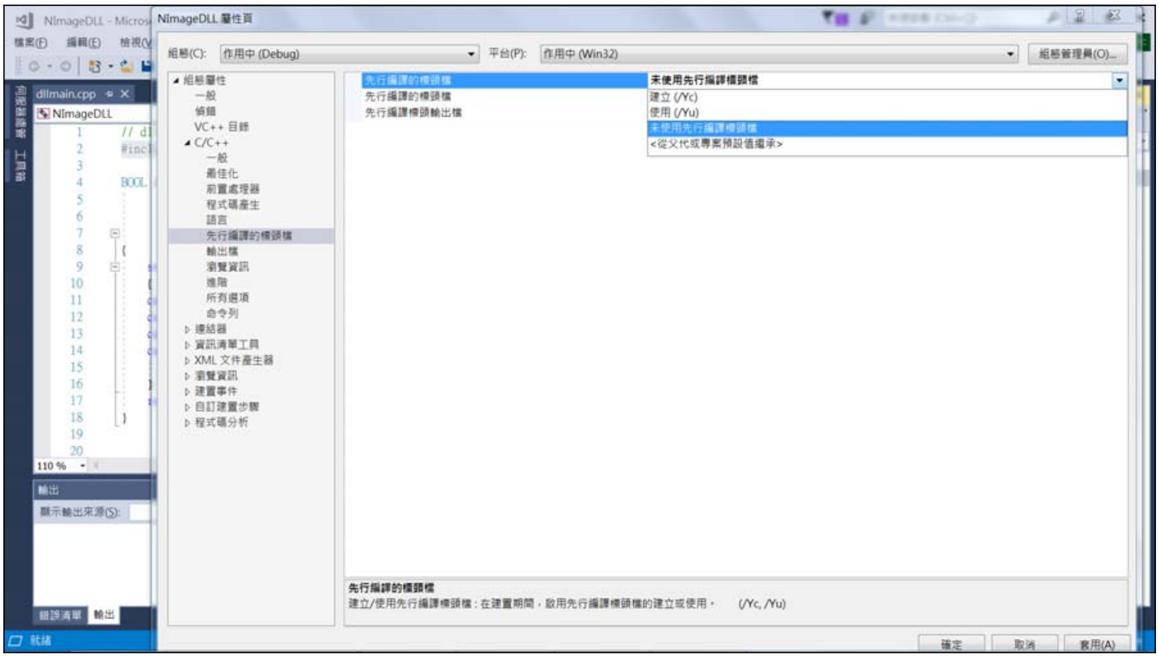
位置(L): C:\Users\Hank\Desktop\Implementation\Test\

方案名稱(M): NImageDLL

☑ 為方案建立目錄(D)

□ 加入原始檔控制中(U)

3. Please follow the steps below to cancel the option of header file pre-compiler. Press **[Alt+Enter]**.



NImageDLL 屬性頁

組態(C): 作用中 (Debug)　平台(P): 作用中 (Win32)　組態管理員(O)...

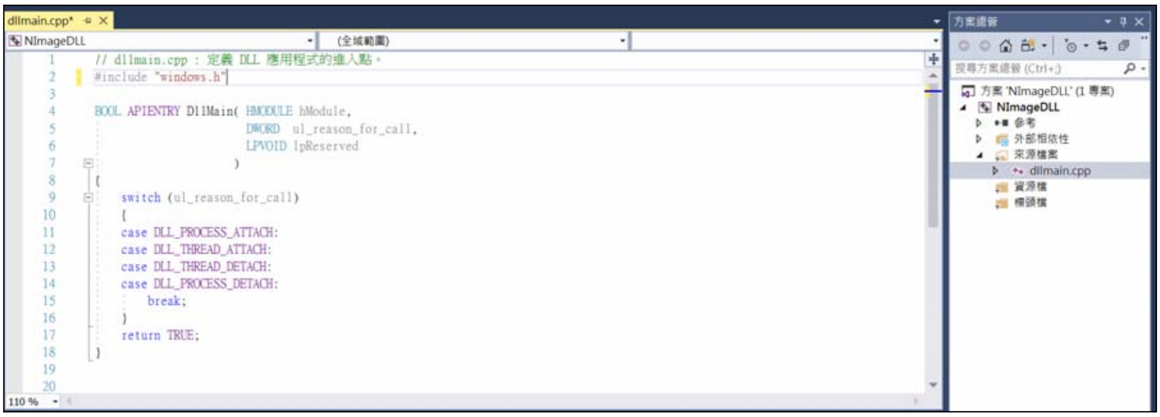
- 組態屬性
  - 一般
  - 偵錯
  - VC++ 目錄
  - C/C++
    - 一般
    - 最佳化
    - 前置處理器
    - 程式碼產生
    - 語言
    - 先行編譯的標頭檔
    - 輸出檔
    - 瀏覽資訊
    - 進階
    - 所有選項
    - 命令列
  - 連結器
  - 資訊清單工具
  - XML 文件產生器
  - 瀏覽資訊
  - 建置事件
  - 自訂建置步驟
  - 程式碼分析

先行編譯的標頭檔: 未使用先行編譯標頭檔
先行編譯的標頭檔
先行編譯標頭輸出檔

建立 (/Yc)
使用 (/Yu)
未使用先行編譯標頭檔
<從父代或專案預設值繼承>

先行編譯的標頭檔
建立/使用先行編譯標頭檔：在建置期間，啟用先行編譯標頭檔的建立或使用。　(/Yc, /Yu)

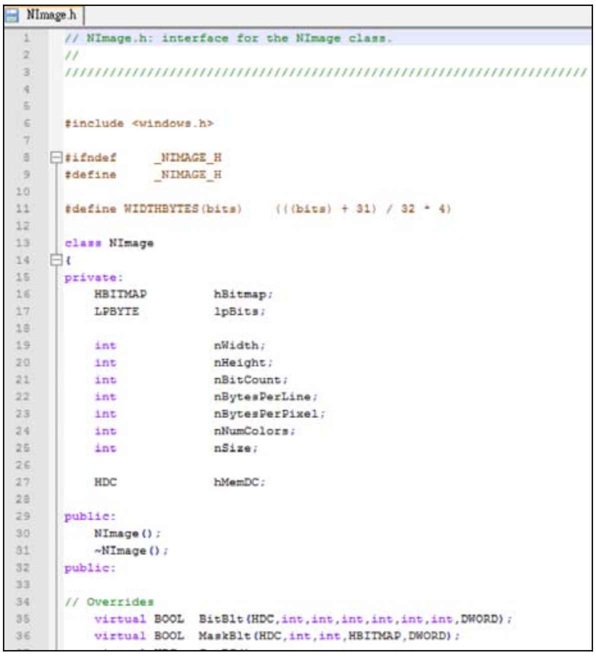
Delete the files of **pch.cpp**, **pch.h** and **framework.h**, and then modify the #include “pch.h” becoming #include <windows.h> in dllmain.cpp as follows.



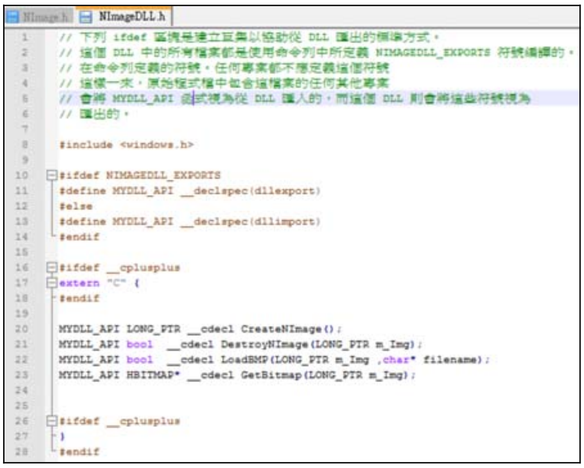
dllmain.cpp*

```
// dllmain.cpp : 定義 DLL 應用程式的進入點。
#include "windows.h"

BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
    case DLL_THREAD_ATTACH:
    case DLL_THREAD_DETACH:
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}
```

4. If you want the image functions to be callable from a DLL on Windows, you must explicitly mark its declaration. The following code **[NImage.h]** and **[NImageDLL.h]** provides a simple demonstration of this.

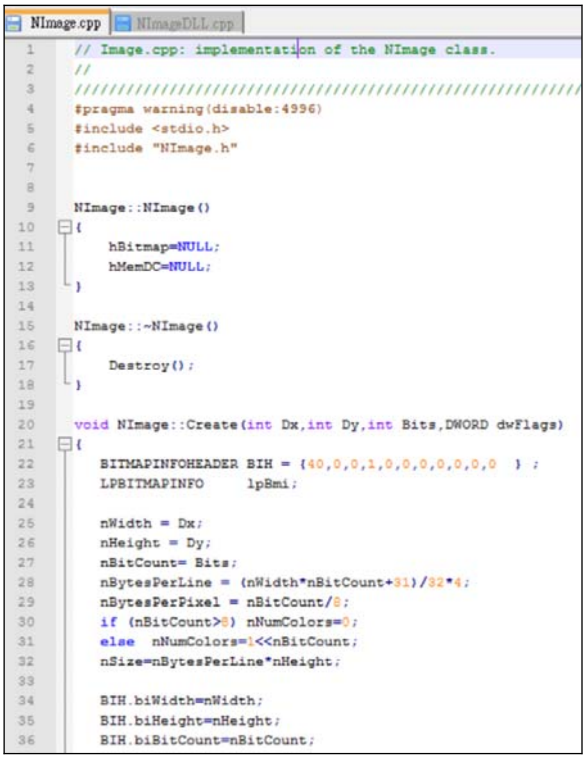


```
// NImage.h: interface for the NImage class.
//
///////////////////////////////////////////////////////////////////////


#include <windows.h>

#ifndef     _NIMAGE_H
#define     _NIMAGE_H

#define WIDTHBYTES(bits)    (((bits) + 31) / 32 * 4)

class NImage
{
private:
    HBITMAP         hBitmap;
    LPBYTE          lpBits;

    int             nWidth;
    int             nHeight;
    int             nBitCount;
    int             nBytesPerLine;
    int             nBytesPerPixel;
    int             nNumColors;
    int             nSize;

    HDC             hMemDC;

public:
    NImage();
    ~NImage();
public:

// Overrides
    virtual BOOL  BitBlt(HDC,int,int,int,int,int,int,DWORD);
    virtual BOOL  MaskBlt(HDC,int,int,HBITMAP,DWORD);
```

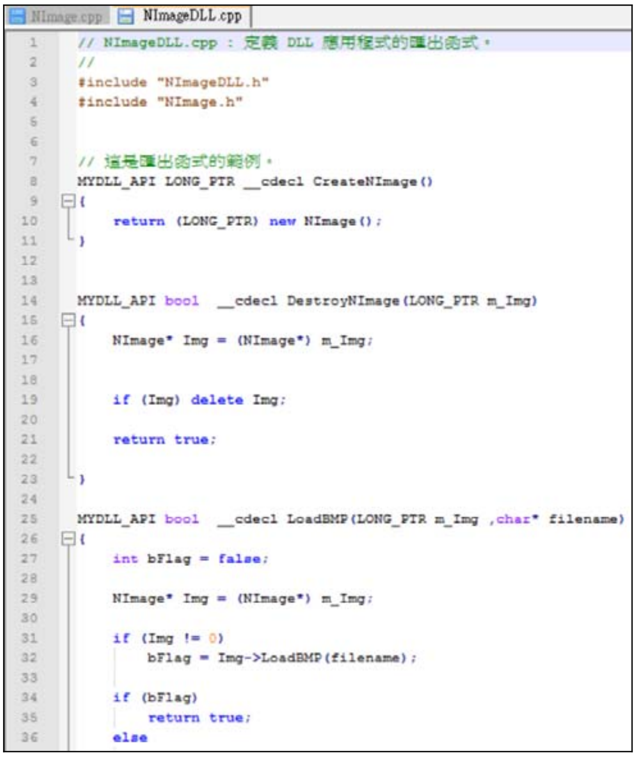


```
// 下列 ifdef 區塊是建立巨集以協助從 DLL 匯出的標準方式。
// 這個 DLL 中的所有檔案都是使用命令列中所定義 NIMAGEDLL_EXPORTS 符號編譯的。
// 在命令列定義的符號。任何專案都不應定義這個符號
// 這樣一來，原始程式檔中包含這檔案的任何其他專案
// 會將 MYDLL_API 函式視為從 DLL 匯入的，而這個 DLL 則會將這些符號視為
// 匯出的。

#include <windows.h>

#ifdef NIMAGEDLL_EXPORTS
#define MYDLL_API __declspec(dllexport)
#else
#define MYDLL_API __declspec(dllimport)
#endif

#ifdef __cplusplus
extern "C" {
#endif

MYDLL_API LONG_PTR __cdecl CreateNImage();
MYDLL_API bool   __cdecl DestroyNImage(LONG_PTR m_Img);
MYDLL_API bool   __cdecl LoadBMP(LONG_PTR m_Img ,char* filename);
MYDLL_API HBITMAP* __cdecl GetBitmap(LONG_PTR m_Img);


#ifdef __cplusplus
}
#endif
```

5. You can then add new or existing source files **[NImage.cpp]** and **[NImageDLL.cpp]** to your project under the Source Files folder in the right-hand pane.

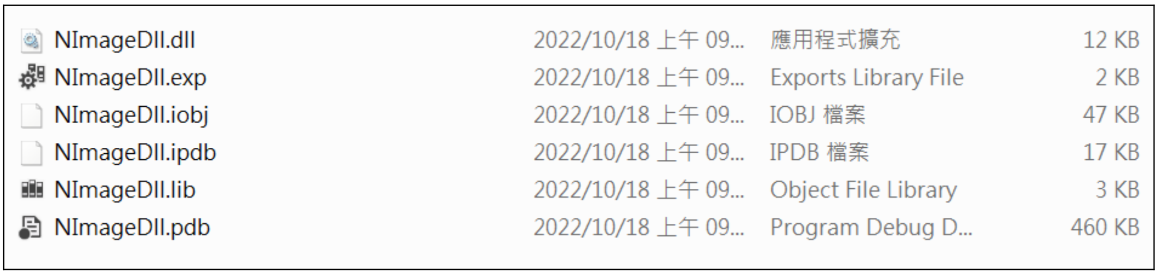


```
// Image.cpp: implementation of the NImage class.
//
////////////////////////////////////////////////////////////////
#pragma warning(disable:4996)
#include <stdio.h>
#include "NImage.h"


NImage::NImage()
{
    hBitmap=NULL;
    hMemDC=NULL;
}

NImage::~NImage()
{
    Destroy();
}

void NImage::Create(int Dx,int Dy,int Bits,DWORD dwFlags)
{
   BITMAPINFOHEADER BIH = {40,0,0,1,0,0,0,0,0,0,0  } ;
   LPBITMAPINFO     lpBmi;

   nWidth = Dx;
   nHeight = Dy;
   nBitCount= Bits;
   nBytesPerLine = (nWidth*nBitCount+31)/32*4;
   nBytesPerPixel = nBitCount/8;
   if (nBitCount>8) nNumColors=0;
   else  nNumColors=1<<nBitCount;
   nSize=nBytesPerLine*nHeight;

   BIH.biWidth=nWidth;
   BIH.biHeight=nHeight;
   BIH.biBitCount=nBitCount;
```



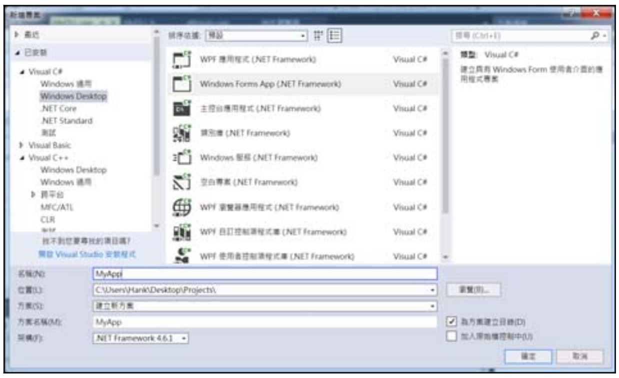
```
// NImageDLL.cpp : 定義 DLL 應用程式的匯出函式。
//
#include "NImageDLL.h"
#include "NImage.h"


// 這是匯出函式的範例。
MYDLL_API LONG_PTR __cdecl CreateNImage()
{
    return (LONG_PTR) new NImage();
}


MYDLL_API bool  __cdecl DestroyNImage(LONG_PTR m_Img)
{
    NImage* Img = (NImage*) m_Img;


    if (Img) delete Img;

    return true;

}

MYDLL_API bool  __cdecl LoadBMP(LONG_PTR m_Img ,char* filename)
{
    int bFlag = false;

    NImage* Img = (NImage*) m_Img;

    if (Img != 0)
        bFlag = Img->LoadBMP(filename);

    if (bFlag)
        return true;
    else
```

6. Build → Build Project (or **Build NImageDLL**), and then Visual Studio will generate a **.dll file** and an associated **.lib import file**.

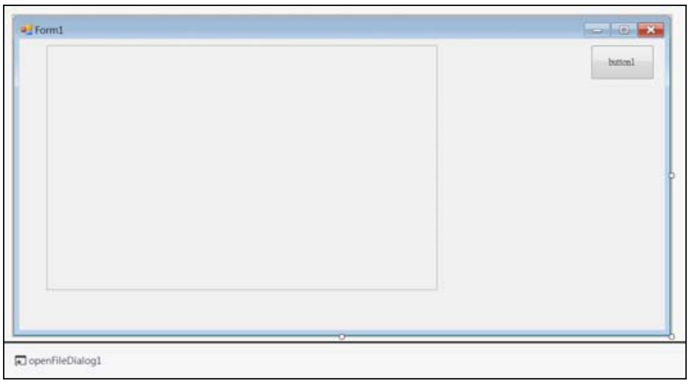


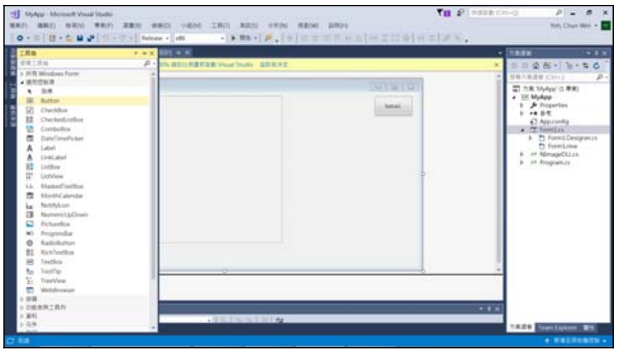
NImageDll.dll	2022/10/18 上午 09...	應用程式擴充	12 KB
NImageDll.exp	2022/10/18 上午 09...	Exports Library File	2 KB
NImageDll.iobj	2022/10/18 上午 09...	IOBJ 檔案	47 KB
NImageDll.ipdb	2022/10/18 上午 09...	IPDB 檔案	17 KB
NImageDll.lib	2022/10/18 上午 09...	Object File Library	3 KB
NImageDll.pdb	2022/10/18 上午 09...	Program Debug D...	460 KB

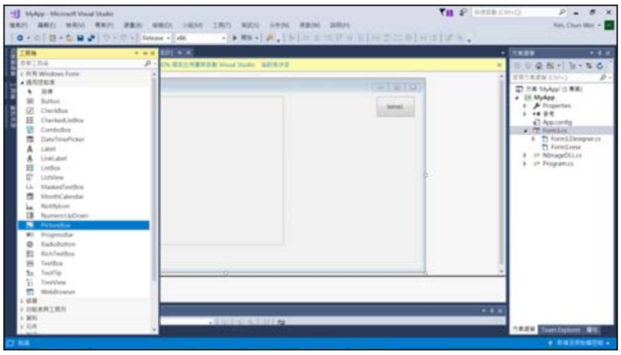
7. The man-machine interface, named MyApp, will be created by using C# to new a **[Button]**, a **[PictureBox]** and a **[OpenFileDialog]** as follows.

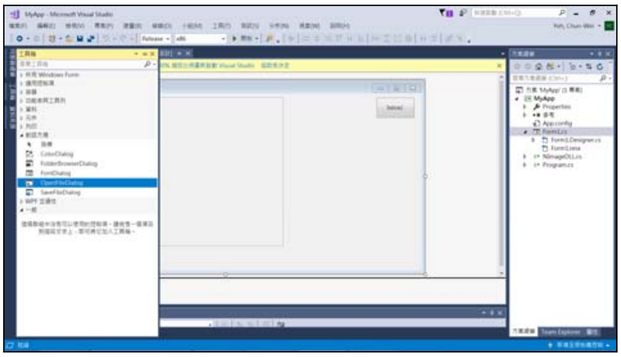


名稱(N): MyApp
位置(L): C:\Users\Hank\Desktop\Projects\
方案(S): 建立新方案
方案名稱(M): MyApp
架構(F): .NET Framework 4.6.1
☑ 為方案建立目錄(D)
□ 加入原始檔控制中(U)

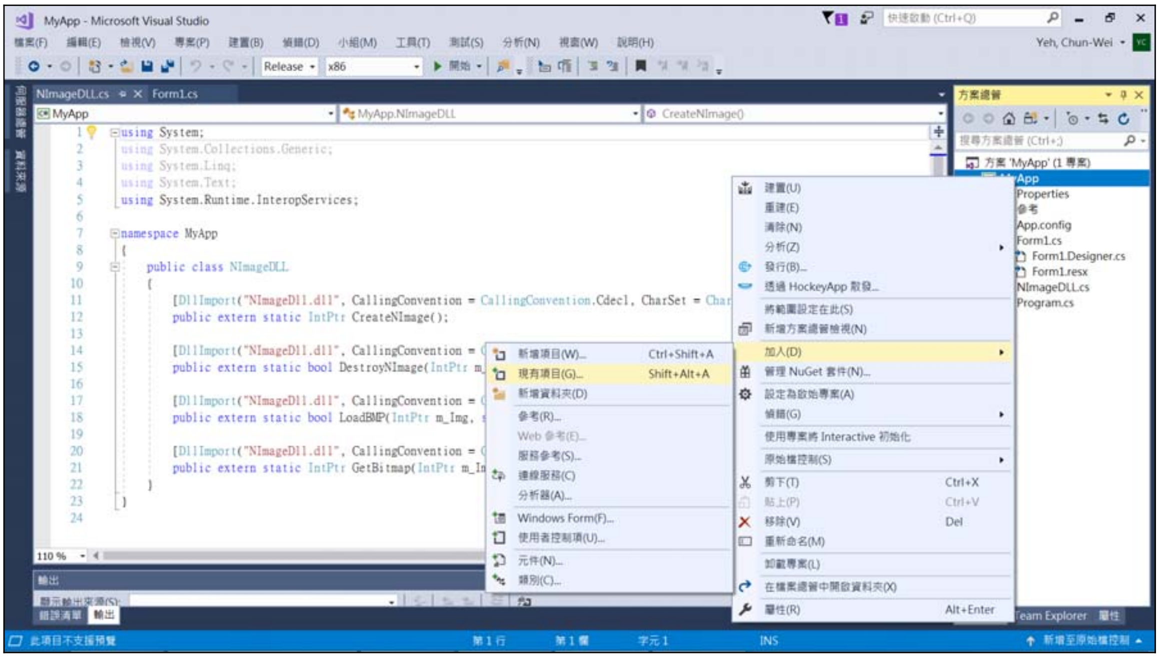






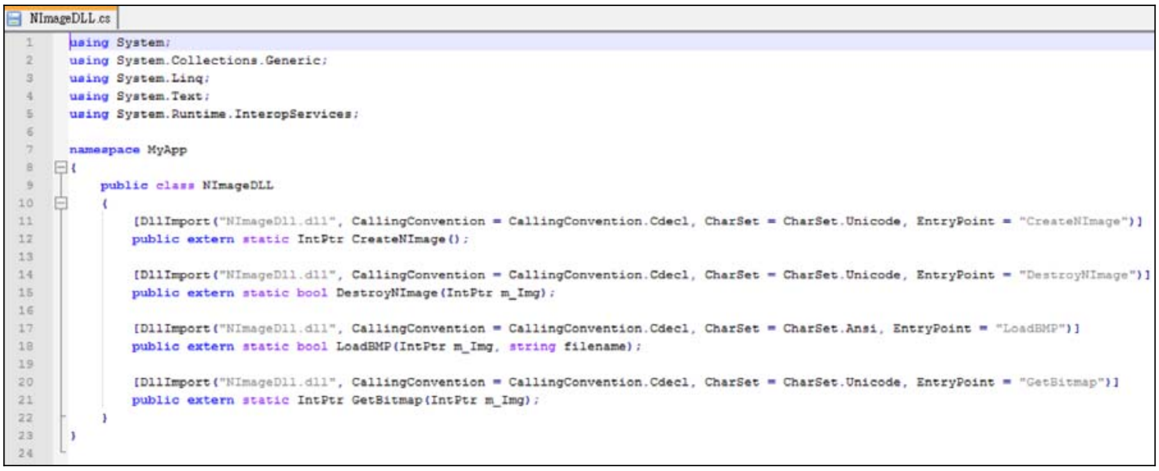


8. Add **[NImageDLL.cs]** to MyApp C# project.



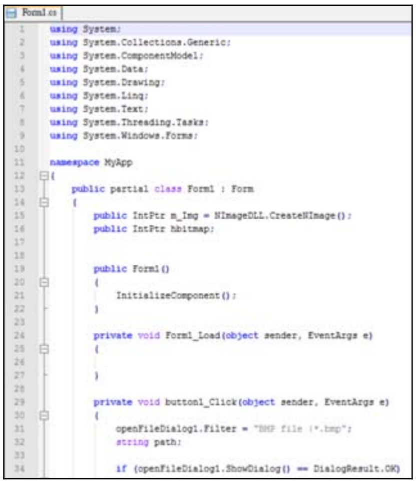
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Runtime.InteropServices;

namespace MyApp
{
    public class NImageDLL
    {
        [DllImport("NImageDll.dll", CallingConvention = CallingConvention.Cdecl, CharSet = Char
        public extern static IntPtr CreateNImage();

        [DllImport("NImageDll.dll", CallingConvention = (
        public extern static bool DestroyNImage(IntPtr m_

        [DllImport("NImageDll.dll", CallingConvention = (
        public extern static bool LoadBMP(IntPtr m_Img, s

        [DllImport("NImageDll.dll", CallingConvention = (
        public extern static IntPtr GetBitmap(IntPtr m_In
    }
}
```

9. The details of **[NImageDLL.cs]** are as follows.

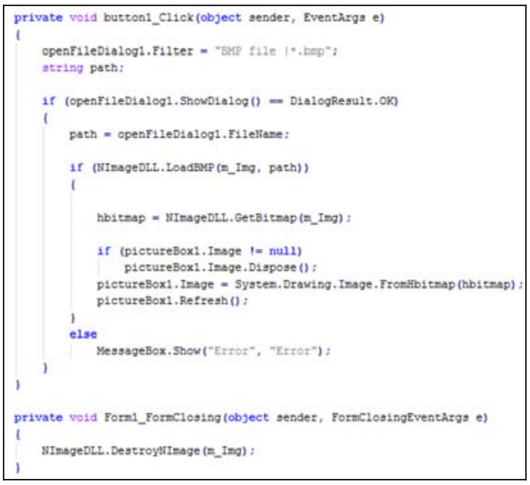


```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Runtime.InteropServices;

namespace MyApp
{
    public class NImageDLL
    {
        [DllImport("NImageDll.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Unicode, EntryPoint = "CreateNImage")]
        public extern static IntPtr CreateNImage();

        [DllImport("NImageDll.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Unicode, EntryPoint = "DestroyNImage")]
        public extern static bool DestroyNImage(IntPtr m_Img);

        [DllImport("NImageDll.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Ansi, EntryPoint = "LoadBMP")]
        public extern static bool LoadBMP(IntPtr m_Img, string filename);

        [DllImport("NImageDll.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Unicode, EntryPoint = "GetBitmap")]
        public extern static IntPtr GetBitmap(IntPtr m_Img);
    }
}
```

10. Call the functions in the events of Form1_FormClosing and buttion1_Click:

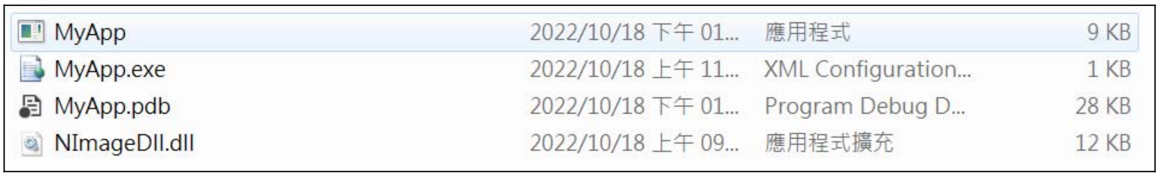


```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace MyApp
{
    public partial class Form1 : Form
    {
        public IntPtr m_Img = NImageDLL.CreateNImage();
        public IntPtr hbitmap;


        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {

        }

        private void button1_Click(object sender, EventArgs e)
        {
            openFileDialog1.Filter = "BMP file |*.bmp";
            string path;

            if (openFileDialog1.ShowDialog() == DialogResult.OK)
```

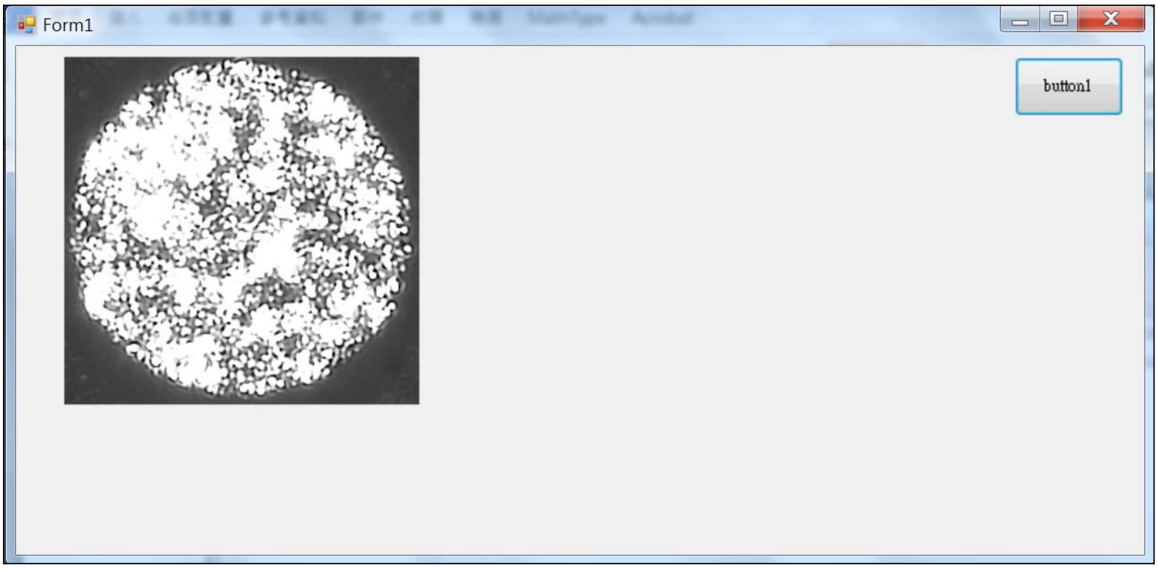


```
private void button1_Click(object sender, EventArgs e)
{
    openFileDialog1.Filter = "BMP file |*.bmp";
    string path;

    if (openFileDialog1.ShowDialog() == DialogResult.OK)
    {
        path = openFileDialog1.FileName;

        if (NImageDLL.LoadBMP(m_Img, path))
        {

            hbitmap = NImageDLL.GetBitmap(m_Img);

            if (pictureBox1.Image != null)
                pictureBox1.Image.Dispose();
            pictureBox1.Image = System.Drawing.Image.FromHbitmap(hbitmap);
            pictureBox1.Refresh();
        }
        else
            MessageBox.Show("Error", "Error");
    }
}

private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
    NImageDLL.DestroyNImage(m_Img);
}
```

11. Build → **Build MyApp**, and copy the **[NImageDll.dll]** file built in step 6 to the folder of MyApp.exe. Now your MyApp.exe is executable to load an image.



MyApp	2022/10/18 下午 01...	應用程式	9 KB
MyApp.exe	2022/10/18 上午 11...	XML Configuration...	1 KB
MyApp.pdb	2022/10/18 下午 01...	Program Debug D...	28 KB
NImageDll.dll	2022/10/18 上午 09...	應用程式擴充	12 KB



Exercises:

1. Run the demo to verify your work.