Using GameStudio in Visual Studio

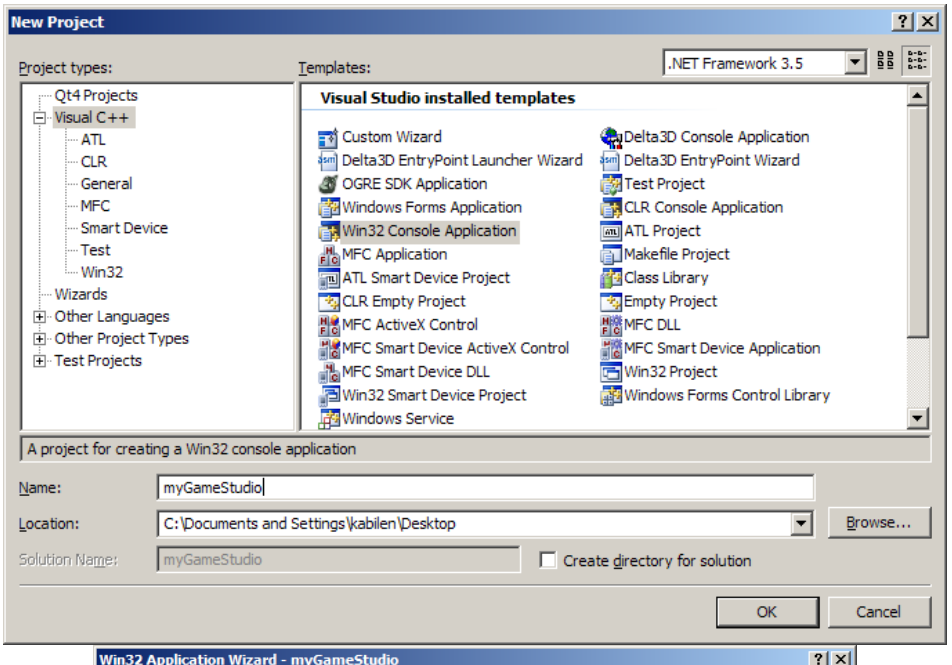
For Professional & Express Editions

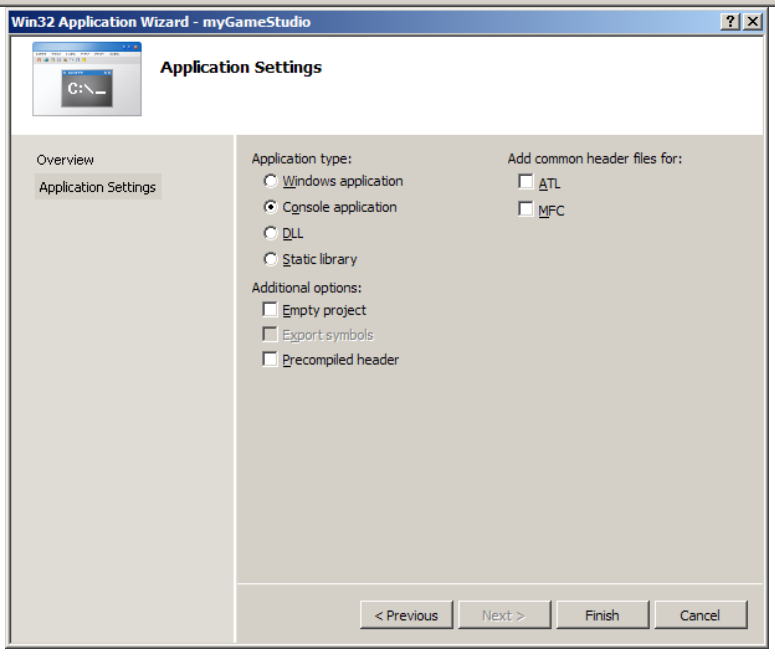
Pre-requisites

- Make sure the platform SDK is installed if using Visual Studio Express.
- Visual Studio Express and the platform SDK can be downloaded freely from Microsoft downloads.

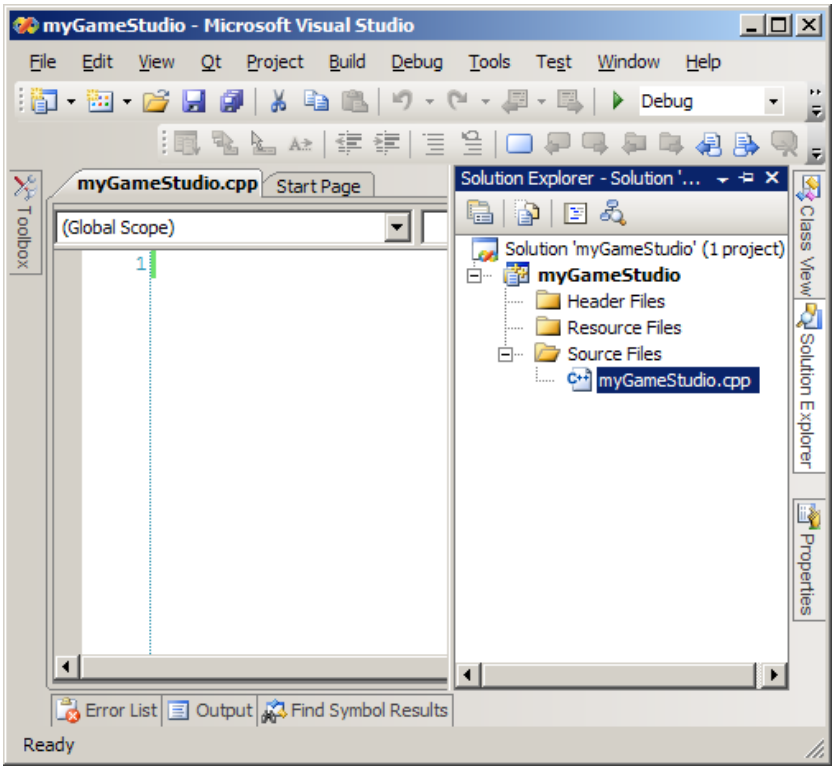
Step 1: Creating Project

- Create a win32 console application project and give it a name e.g. myGameStudio



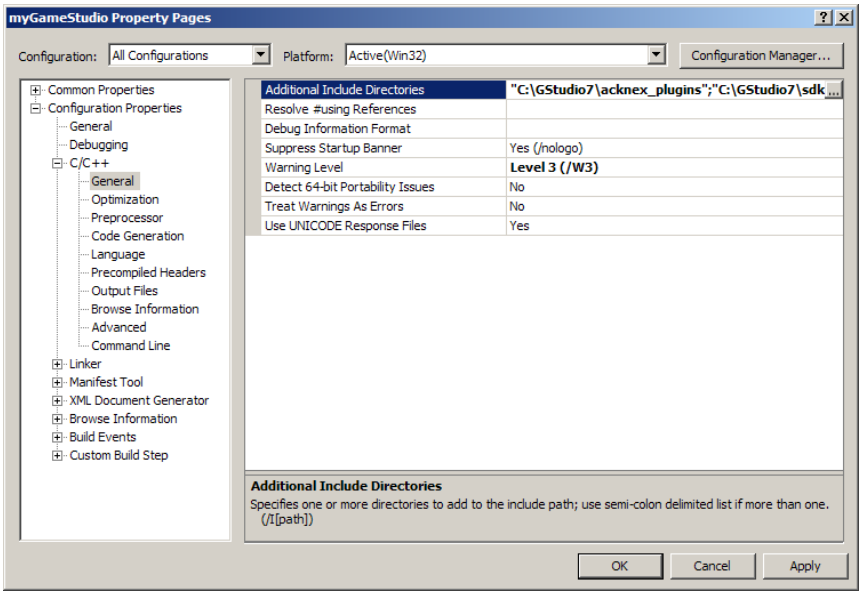


- 4 files will be created in the solution explorer: **stdafx.h**, **targetver.h**, **myGameStudio.cpp** and **stdafx.cpp**.
- Remove all files and keep only myGameStudio.cpp, and clear the code in **myGameStudio.cpp** as shown in the figure below:

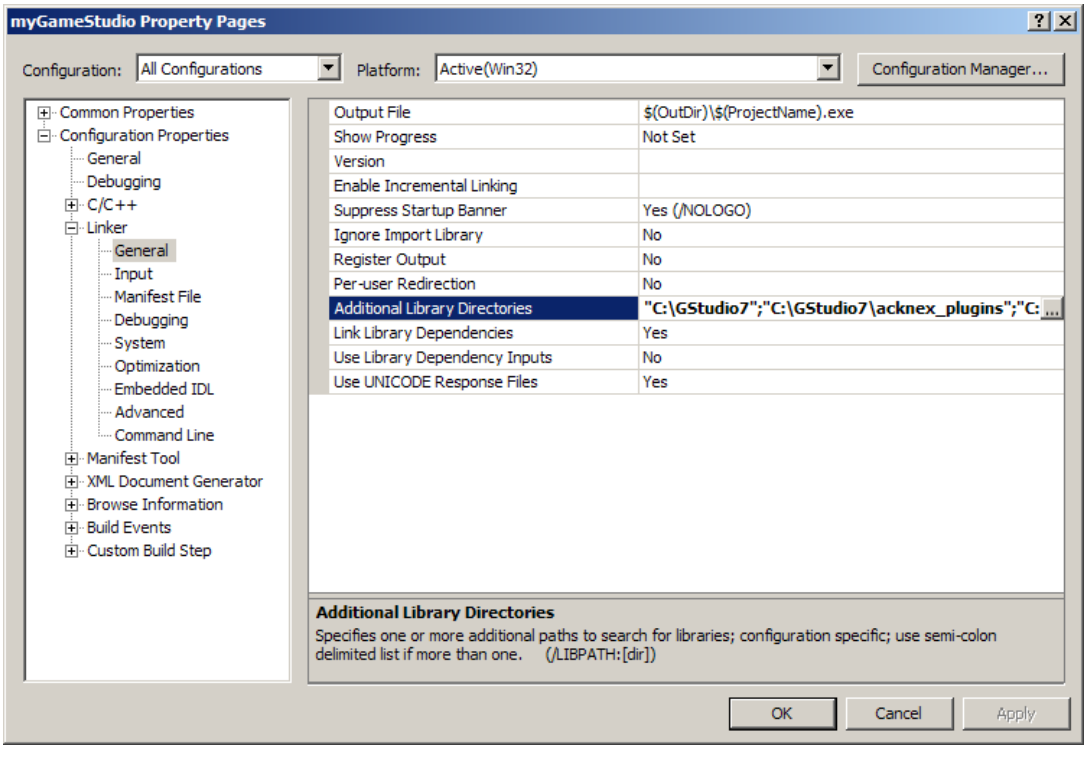


Step 2: Project Settings

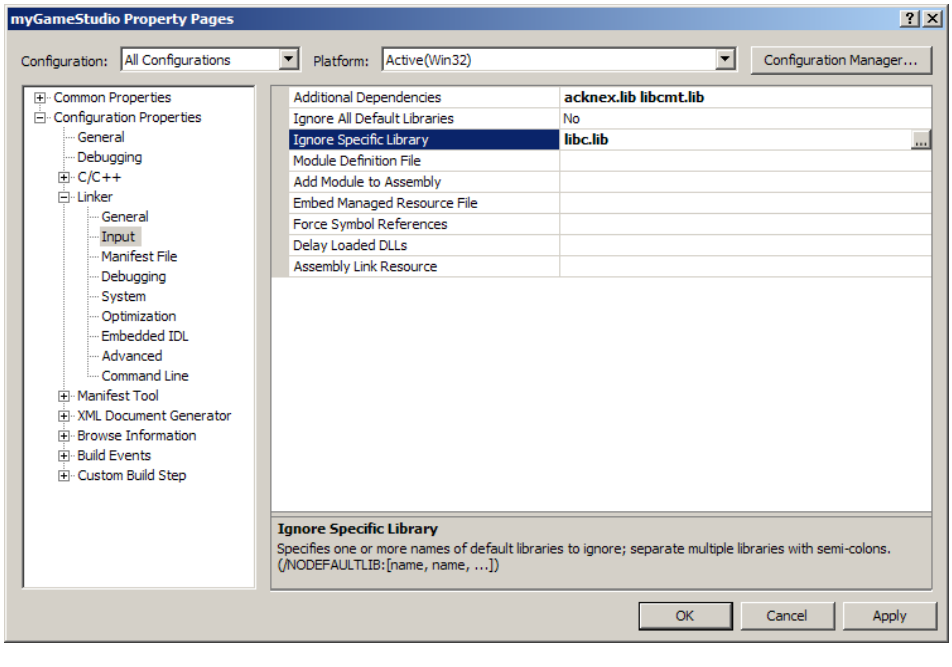
- Edit the project settings: Add **"C:\GStudio7\acknex_plugins";"C:\GStudio7\sdk_engine"** to additional include directories. (Assuming your gamestudio is installed at C:\GStudio7)



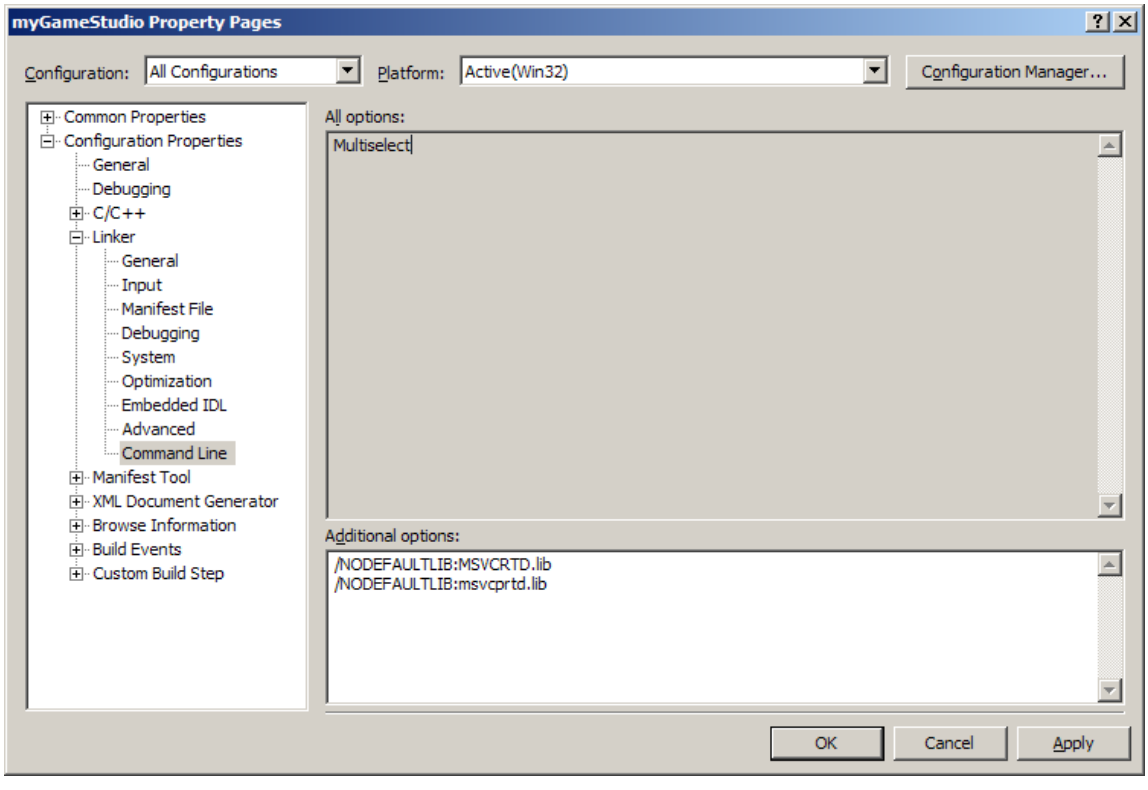
- Under Linker, add the following Additional Library Directories: **"C:\GStudio7";"C:\GStudio7\acknex_plugins";"C:\GStudio7\sdk_engine"**



- Under Linker->Input, Addional dependencies, add **acknex.lib libcmt.lib.** And under Ignore Specific Library, add **libc.lib**



- Under Linker->Command Line, add the following to the additional options:
  /NODEFAULTLIB:MSVCRTD.lib
  /NODEFAULTLIB:msvcprtd.lib



- That's all for the linking.

Step 3: Creating a sample project

- Open the file myGameStudio.cpp and add the following code to it: (note: assume you have built a level named as arena.wmb)

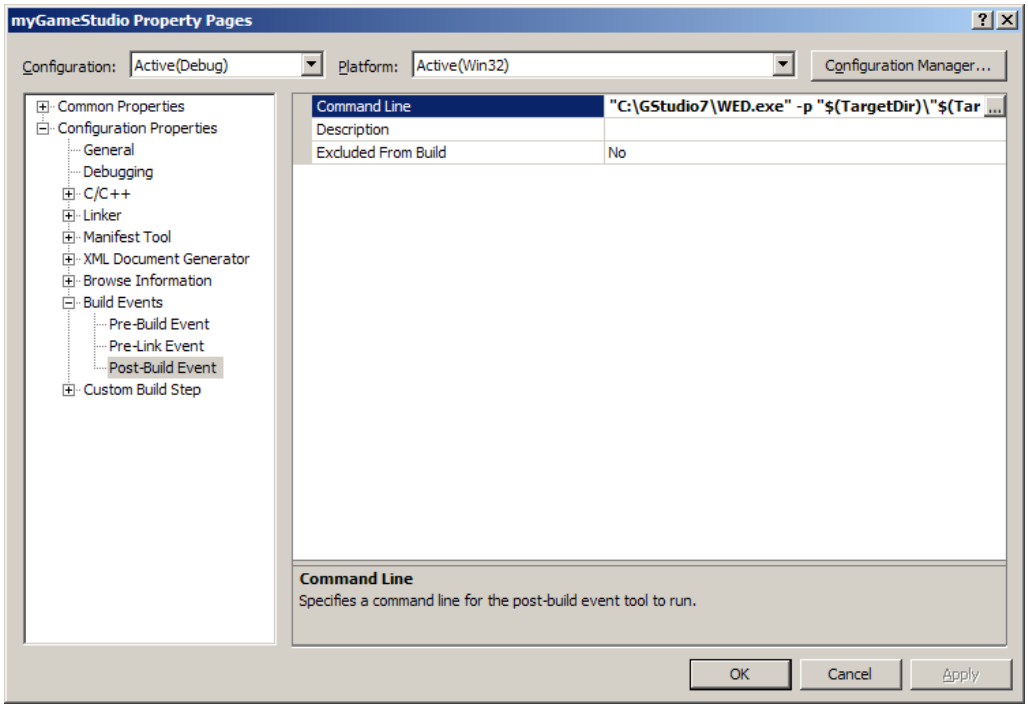
```
//#define WIN32_LEAN_AND_MEAN
#include <windows.h>
#include <stdlib.h>
#include <malloc.h>
#include <memory.h>
#include <tchar.h>

// Include the engine data types, variables, and functions
#include "adll.h"

int main()
{
      engine_open("arena.wmb");
    while (engine_frame());
      engine_close();
      return 0;
}
```

- Compile the project. Press F5.
- This will create a myGameStudio.exe file in the debug/release folder of the VC++ project.
- To execute this exe, we need an acknex.dll which can be generated from WED.
- The following are the ways to run the exe:
    - Copy the generated myGameStudio.exe and arena.wmb into C:\GStudio7 where WED.exe resides. There you can run the exe.
    - Another way is to generate an acknex.dll with WED while publishing. Copy the arena.wmb, the acknex.dll and the myGameStudio.exe into one folder. Then can distribute.
    - To automate the step above, in the Visual Studio project settings add the following to the post-build event:

      "C:\GStudio7\WED.exe" -p "$(TargetDir)\"$(TargetFileName)



    - In this way, every time you build your project in Visual Studio, an acknex.dll is created from WED through a command line and is placed in your target directory, i.e. the debug or release folder.

Step 4: Try one of the Tutorial

```
#define WIN32_LEAN_AND_MEAN
#include <windows.h>
#include <stdlib.h>
#include <malloc.h>
#include <memory.h>
#include <tchar.h>
#include "adll.h"

ENGINE_VARS *ev;

int main()
{


      ev = engine_open(NULL);
      if (!ev) return 1;      // acknex.dll not found

      level_load("arena.wmb");

    SETV(fps_max,50);


      while (engine_frame())
      {


            v(camera).pan += 3 * v(key_force).x;
            v(camera).tilt += 3 * v(key_force).y;



            VECTOR vMove;
            vMove.x = 6.0 * (v(key_w) - v(key_s));
            vMove.y = 6.0 * (v(key_a) - v(key_d));
            vMove.z = 0;

            vec_rotate (&vMove, (ANGLE*)&v(camera).pan);
            vec_add ((VECTOR*)&v(camera).x, &vMove);
      }


      engine_close();
      return 0;
}
```

Special Thanks to: Pararealist pararealist@ntlworld.com for helping in the successful conceptualizing of this tutorial.