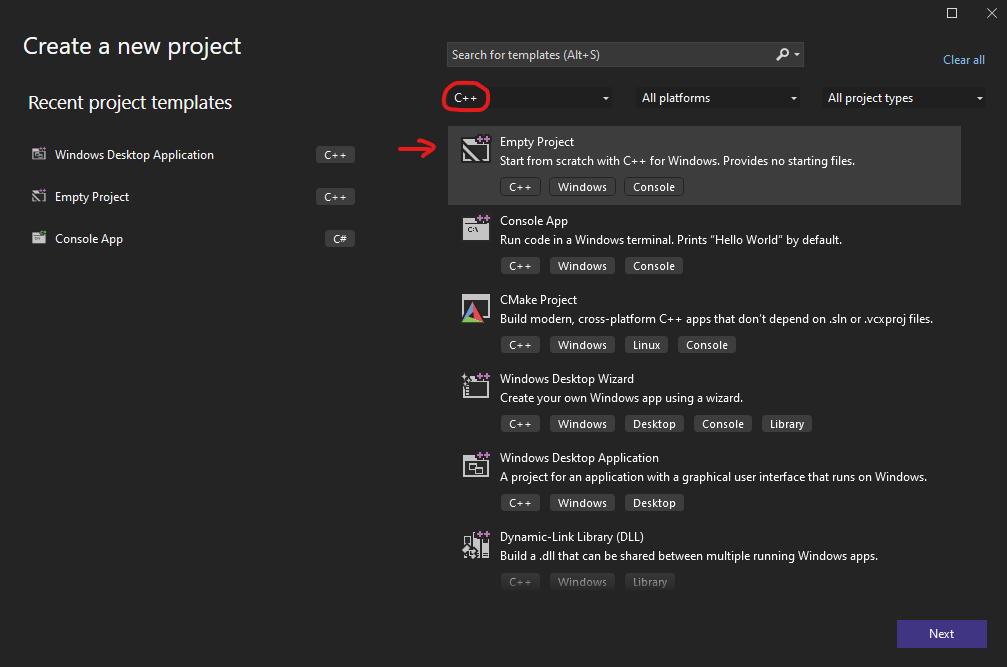
**Example – Porting a project from old XP system to Windows 10.**

The old codes were written in 32-bit format on MS visual c++ 6 on windows XP PCs. We want to port the old codes to work on MS visual c++ 2022 community on a 64-bit windows 10 PC in this example.

**Step 1 – Creating an empty project:**

Create a new empty project in MS VC++ 2022.



The location of the project must be C:\robotcode\projects\

**Step 2 – Project configurations:**

Change solution platform from x64 to x86. Then open project properties and change the following configurations:

1. Project -> Properties -> Configuration Properties -> Advanced -> Character set: set to “Use multi-byte character set”
2. Project -> Properties -> Configuration Properties -> VC++ Directories -> Include Directories: add the include (header files) to the include directories. Must be found in the address: C:\robotcode\include
3. Project -> Properties -> Configuration Properties -> VC++ Directories -> Source Directories: add the source (.cpp files) to the source directories. Must be found in the address: C:\robotcode\source
4. Project -> Properties -> Configuration Properties -> Linker -> System -> SubSystem: Change to “Windows(/SUBSYSTEM:WINDOWS)” (because this a windows application not a console application).

A screenshot of a computer

Description automatically generated with medium confidence

**Step 3 – Adding the project source and header files**

Create your project’s main source (.cpp) and header (.h) files. Choose a template project you want to make your code based on. Open the main source (.cpp) and header (.h) files of the template project and copy & paste the codes to yours.

**Step 4 – Ignoring the MS VC++** sprintf\_s() **and** sscanf\_s()**warnings**

The original library and codes use sprintf() and sscanf() functions. However, these functions are forced to be replaced with sprintf\_s()and sscanf\_s()by the MS VC++ compiler. To ignore this error, do the following:

Project -> Properties -> Configuration Properties -> C/C++ -> Preprocessor -> Preprocessor definitions: Add \_CRT\_SECURE\_NO\_WARNINGS and click OK.

**Step 5 – Add the project dependencies**

By right clicking on your project name in the “solution explorer” and selecting “add -> existing item”, add all the project dependencies from “C:\robotcode\include”, “C:\robotcode\source” to your project.

Note: If you don’t add any of the project dependencies, you will face errors saying “unresolved external symbol”. Here is an example of how the errors might look like:

A computer screen capture

Description automatically generated with medium confidence

**Step 6 – Create a data folder**

Create a temp data folder for your experiment in C:\data\<your project name>\

The code does not make this folder for you and your experiment might not run.

**Step 7 – Change the paths in your main file**

There are some paths in the main source codes of projects that might need change especially if you are using a new PC. Some of the paths to check in your main code are brought here.

-If you are using sounds for your task:

string TASKSOUNDS[8] = {

"C:/robotcode/util/wav/ding.wav", // 0

"C:/robotcode/util/wav/smb\_coin.wav", // 1

"C:/robotcode/util/wav/chimes.wav", // 2

"C:/robotcode/util/wav/smb\_kick.wav", // 3

"C:/robotcode/util/wav/bump.wav", // 4

"C:/robotcode/util/wav/chord.wav", // 5

"C:/robotcode/util/wav/smb\_pipe.wav", // 6

"C:/robotcode/util/wav/error.wav" // 7

};

-In the WinMain which is your main program:

gExp = new MyExperiment("eir2", "eir2", "C:/data/ExtrinsicIntrinsicRepetition/eir2/");

// initialize s626cards

s626.init("c:/robotcode/calib/s626\_single.txt");

**Step 8 – Change the experiment setting if needed**

-In the WinMain which is your main program:

gScreen.init(gThisInst, 1920, 0, 1920, 1080, &(::updateGraphics));

The first two numbers are the relative position of the second monitor in pixels and the second two are the resolution in pixels.

**Step 9 – Don’t forget the target files**

Add the experiment target folder (including .tgt files) to the main folder of your project where your main .cpp and .h files are (e.g. C:\robotcode\projects\ExtFlxChord\ExtFlxChord).

**Extra Notes:**

Your code should compile and build if you do all the steps carefully! If it still doesn’t, you might be using some source and include files or a template project that were not ported by “Ali Ghavampour” because he didn’t need them for his project. You might need to fix them. If that was the case, take a look at the “Codes ChangeLog” document and please add your final changes to the log file for future reference.