Compiling and Debugging the SDK on Visual Studio 2005 Express

Those are slightly modified instructions from Chazcon. I added Screenshots of the key interface elements. Some changes have to be made to allow debugging. I will keep Chazcons steps and mark up the key changes to Chazcons tutorial, so those familiar with it can just read over the changes.

Credit goes to Chazcon for his Compiling-tutorial, to Jean Elcard for the Makefile i used as base and to DaveMcW who's gave the key peace of information as to why debugging was not working in the first place.

Quote Chazcon:

Credit goes to DaveMcW, Kael, and Maian for doing all the hard work. I've only combined their efforts into one post.

If you need more background you can check out the compiler threads on CivFanatics - all the information needed was collected from those 2:
By Kael: http://forums.civfanatics.com/showthread.php?t=166933
By Chazcon: http://forums.civfanatics.com/showthread.php?t=196283

Installing the Tools

This part is unchanged from Chazcons tutorial.

- Download and install Microsoft Visual C++ Toolkit 2003
http://kael.civfanatics.net/files/VCToolkitSetup.exe
- Download the following three library files (msvcrt.lib, msvcrtd.lib, msvcprt.lib and put them in the folder:
C:\Program Files\Microsoft Visual C++ Toolkit 2003\lib
http://kael.civfanatics.net/files/msvcrt.lib
http://kael.civfanatics.net/files/msvcrtd.lib
http://kael.civfanatics.net/files/msvcprt.lib
- Download and install the Microsoft Platform SDK
http://kael.civfanatics.net/files/PSDK-x86.exe
This one might give you slight trouble, as Microsoft took down this exact version. You should still be able to find it somewhere on the net. I use *Microsoft Platform SDK for Windows Server 2003 R2* which was available from MS directly and works fine.
- Download and install Microsoft Visual C++ 2005 Express Edition
http://msdn.microsoft.com/vstudio/express/visualc/download/

Preparing the Working Folder

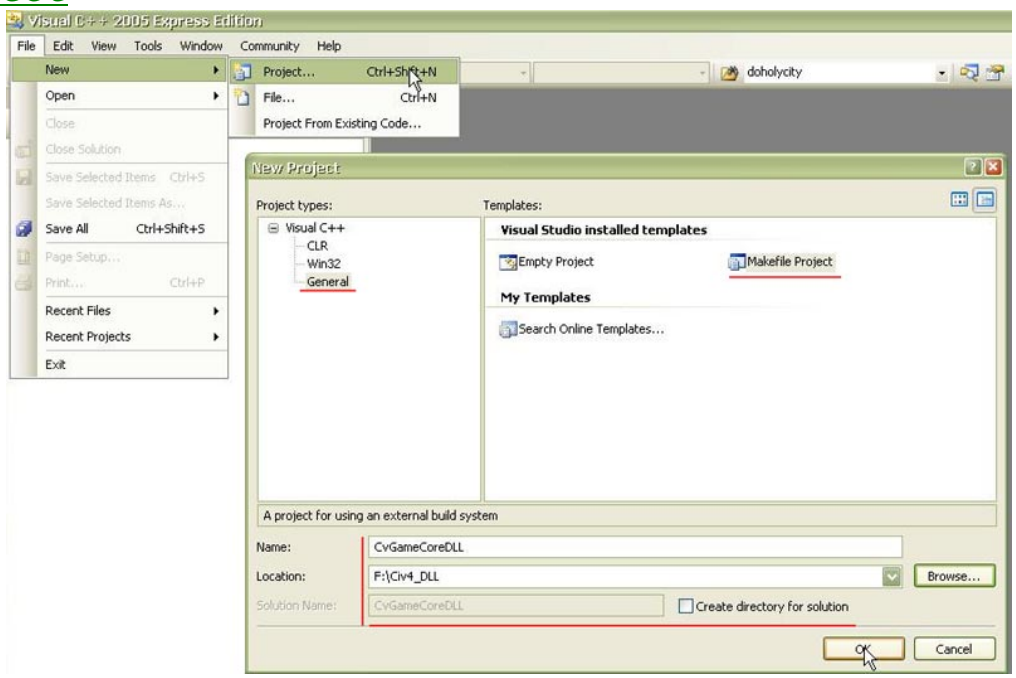
With BtS you will find the most recent SDK Source Files in the folder CvGameCoreDLL.

- Make a copy of this complete folder, including its subfolders to a location where you want to keep your working files.
From now on everything said refers to the working copy of the files. Don't ever modify the original files - you might need them.
- Delete the files CvGameCoreDLL.vcproj and CvGameCoreDLL.rc.
- Put the Makefile that came with this instruction in your working folder. (The Makefile will only work on the most recent as of now BtS version 3.17 you should be able to modify it for other versions of the game however, if you need it)
- Open the Makefile in your Code editing program. You can use the freshly installed VS2005 IDE. Or something you prefer. ***Windows notepad does not work well here.***
- At the very top, change the line that begins with TOOLKIT to the path of your Visual C++ Toolkit 2003 directory.
- Change the line that begins with PSDK to the path of your Microsoft Platform SDK.
- Save the Makefile and exit.

Setting up the Project

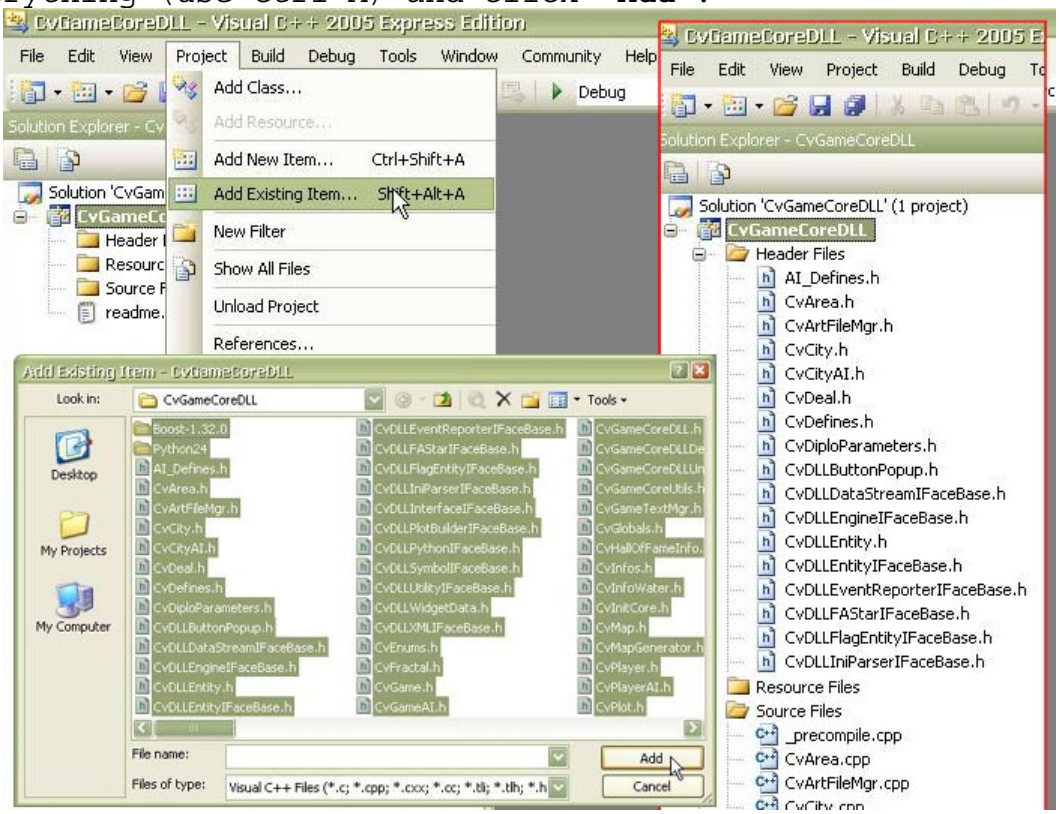
Launch the IDE.
Do not open the existing project file in Visual C++ 2005 Express Edition.

If you followed the instructions above, it should not even exist. If it still does, delete it now.



- Click ***-File- -New- -Project-***. The "New Project" window will open.
- In the "Project types" box select ***-General-***. In the "Templates" box select ***-Makefile Project-***.
- In the "Name" box type "CvGameCoreDLL".
- In the "Location" box browse to the folder where you copied your CvGameCoreDLL folder to. This folder MUST be the last folder listed in this box. For example, if you copied CvGameCoreDLL to another folder like so: *C:\MyModsFolder\CvGameCoreDLL* , then the entry in the "Location" box must read: *C:\MyModsFolder*
- UNCHECK the box that says "Create Directory for Solution".
- Click ***-OK-***. The "Makefile Application Wizard" will open (No picture here). Click ***-Next-***. Clear all the boxes. Click ***-Finish-***.

- Click ***-Project- -Add Existing Item…-***. The "Add Existing Item - CvGameCoreDLL" window will open. This folder will contain all of the source code files, the Boost-1.32.0 folder, and the Python24 folder. Select everything (use Ctrl-A) and click ***-Add-***.



• Click ***-Project- -CvGameCoreDLL Properties-***. The "CvGameCoreDLL Properties Pages" window will open.

• In the left-hand box under "Configuration Properties" select "NMake".

• In the upper-right-hand corner click the "Configuration Manager" button. The "Configuration Manager" window will open.

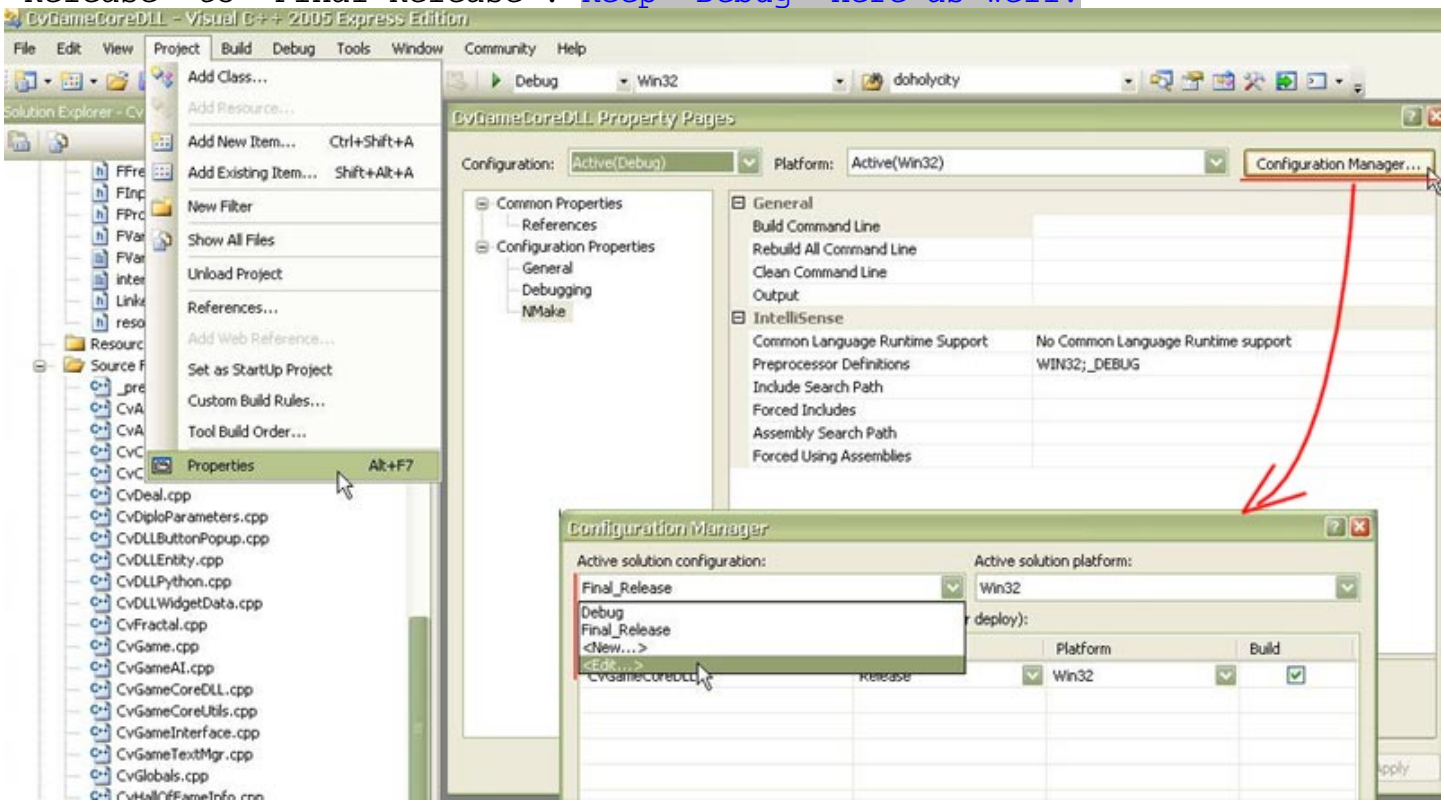
• In the upper-left-hand box named "Active solution configuration", click the drop-down button and select '<Edit…>'. The 'Edit Solution Configurations' window will open.

• Keep the "Debug" Configuration.

• Select "Release" and click the "Rename" button. Change the name from "Release" to "Final_Release". Click "Rename". Click "Yes".

• Close the "Edit Solution Configurations" window.

• In the "Project contexts" window, in the "Configurations" column, click the drop-down button and select "<Edit…>". The "Edit Project Configurations" window will open. Repeat the steps above to rename "Release" to "Final Release". Keep "Debug" here as well.



• Close the "Configuration Manager" window. Now you're back at the "CvGameCoreDLL Properties Pages" window. Make sure in the left-hand box under "Configuration Properties" you've selected "NMake".

• Select the "Final_Release" configuration from the "Configuration:" drop-down-box in the top left corner.

• In the right-hand box under "General" it says "Build Command Line". Click the empty box to the right of it and type *nmake Final_Release*.

• In the next line down, to the right of where it says "Rebuild All Command Line", click in the empty box and then click the button in the box at the right. The "Rebuild All Command Line" window will open. Type *nmake clean_Final_Release*, and then directly below it on a second line, type *nmake Final_Release*. Click ***-OK-***.

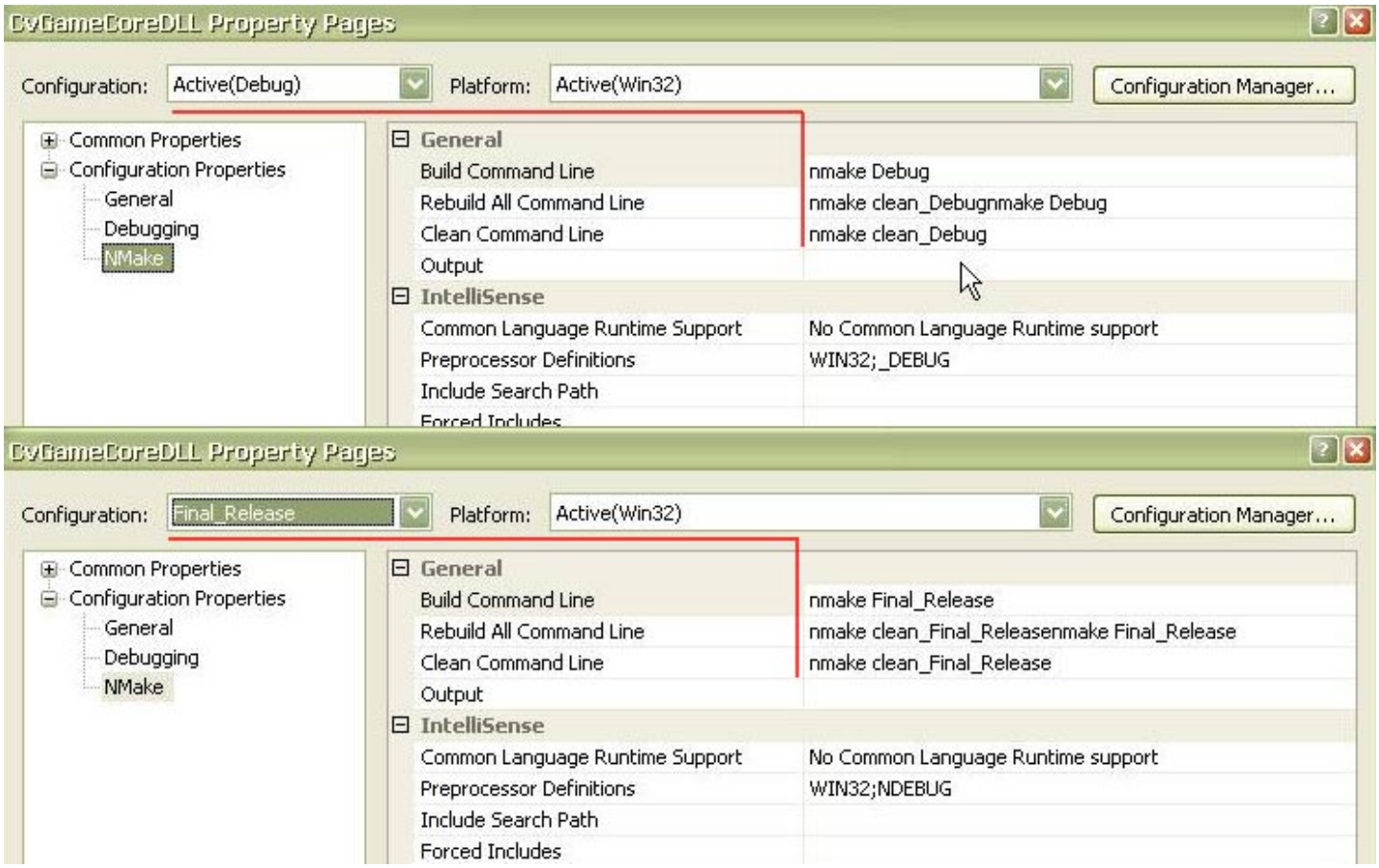
• In the next line down, to the right of where it says "Clean Command Line", type *nmake clean_Final_Release*.

• Select the "Debug" configuration from the "Configuration:" drop-down-box in the top left corner, and repeat the 3 steps above for Debug.

Note the underscores in the clean command.

You want the "… Command Line" boxes under General look like this:

	Final Release	Debug
Build Command Line	nmake Final_Release	nmake Debug
Rebuild All Command Line	nmake clean_Final_Release nmake Final_Release	nmake clean_Debug nmake Debug
Clean Command Line	nmake clean_Final_Release	nmake clean_Debug



Close the configuration windows with -OK-. Save your Project. It will create a new CvGameCoreDLL.vcproj file, which you can open later. It will have your settings saved.

You are done now. For a test you should compile the unchanged code under both Configurations (It takes a bit of time depending on your Hardware), to make sure everything does work as expected. You should use ***Build->Build Solution*** to do that.

You can also compile a single changed file without building the whole project. I did not find a way to make it from the IDE interface, but you can invoke nmake from console. Use Tools->Visual Studio Command Prompt to get a console with all environment variables properly set (You will have to change the directory to where your working folder is).
The target names are not too handy. It's basically just the side effect of having those targets defined as the Builds need them.

nmake Debug/CvUnit.obj
nmake Final_Release/CvUnit.obj
for example will compile just CvUnit.cpp in the Debug or Final_Release configuration.

Running the Debugger

After compiling your project in the Debug configuration, you can run the game using the Debug-able CvGameCoreDLL.dll. It is a good deal bigger than your normal DLL and there is also a big CVGameCoreDLL.pdb file coming with it, which will contain information the debugger might need.

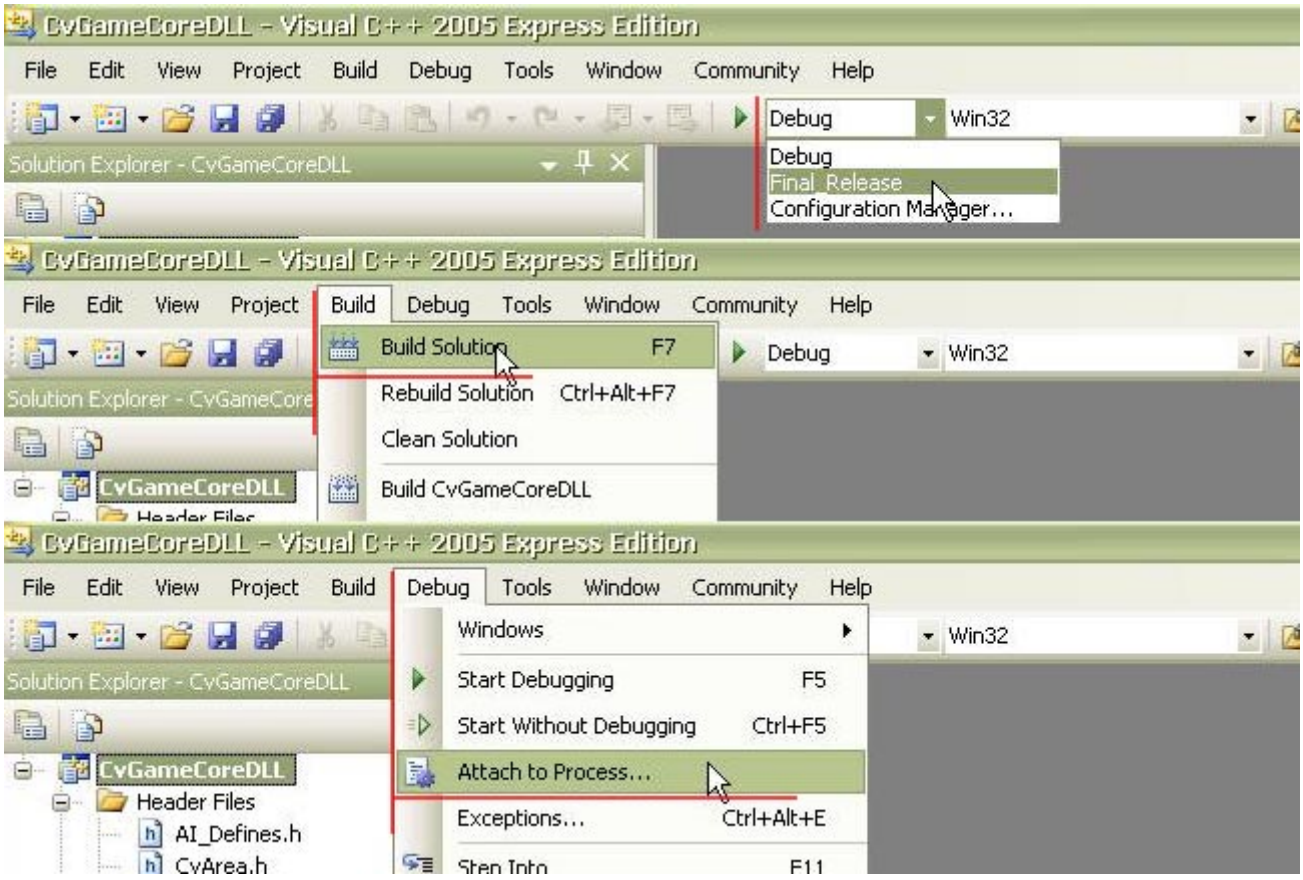
While the IDE is running with your project open, start the Game with the Debug-DLL. Choose ***Debug->Attach To Process…*** from the menu. Choose the Civ4 Process. You might need to re-enable some local Services – on XP debugging with VS2005 Express does require Terminal Services to be running.

I recommend running Civ4 Windowed; otherwise you might not be able to access the IDE after the game freezes on a breakpoint.

In the debugger output window you will see the List of Libraries loaded in the Process. Most of them will say "No Symbols Loaded". This is fine. We are going to debug our CvGameCoreDLL.dll, so this is the only one we need symbols for.

Now you can debug. For testing set a Break-point on some function. CvGame::DoHolyCity() works fine – it is called once per turn. When the Game reaches the Breakpoint it will freeze waiting for your debugging action.

Another good test is a break-point at CvUnit::GetExperience(). When the game stop there, you can access the memory of the Unit instance that called the method and change its Experience level. The unit should be promoted on the next turn.



Have fun moding and debugging.
Refar @ CivFanatics
17.07.2008