Compiling Orx dependencies for new Visual Studio editions

This document is a cheat sheet for compiling up orx dependencies for new editions of Visual Studio in order to contribute back the compiled libraries into the release versions of orx.

This page is not for general use, it is low traffic, and prone to frequent changes and errors. If you are looking for compiled orx dependancies, you can download a precompiled orx, or clone from the the regular repo.

Clone https://bitbucket.org/orx/orx-extern

Build libwebp

1. In a VS2015 console, go to the libwebp folder and: nmake /f Makefile.vc CFG=release-static RTLIBCFG=static OBJDIR=output
2. Ignore errors, locate the lib at: libwebp\output\release-static\x86\lib\webpdecoder.lib
3. Switch to 64 bit compiler mode with: C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\bin\x86_amd64\vcvarsx86_amd64.bat
4. nmake /f Makefile.vc CFG=release-static RTLIBCFG=static OBJDIR=output
5. Ignore errors, locate the lib at: libwebp\output\release-static\x64\lib\webpdecoder.lib

Build OpenAL-Soft

1. Install CMake 2.6 or better
2. In the project root, create a VS2015 project with: cmake -DLIBTYPE=STATIC -G “Visual Studio 14 2015”
3. Open the OpenAL.sln into Visual Studio.
4. Switch to release, right click on the OpenAL32 project and select build to make the 32-bit static lib.
5. Find the 32 bit version in \Release\OpenAL32.lib
6. For 64bit: Select the dropdown that says win32
7. Click and select Configuration Manager
8. In the Active Solution Platform dropdown, click <New>
9. Select x64 and copy from win32 (leave all options default)
10. In the OpenAL32 project properties
11. - In C/C++→Code Generation
12. - - select Mulithreaded (/MT)
13. - In Librarian / Command Line
14. - - remove additional options relating to /machine:X86
15. In the common project properties
16. - In Librarian / Command Line
17. - - remove additional options relating to /machine:X86
18. Right click the OpenAL32 project and select build to make the 64-bit static lib.
19. Find the 64bit version in \openal-soft\x64\Release\OpenAL32.lib

Build freetype

Follow pretty much the same as OpenAL-Soft

Build Box2D

1. Copy build VC13 to VC14
2. Create a lib/msvs2014/32 and 64 folders
3. Open solution is VS2015
4. Allow it to convert the project
5. Convert the solution by selecting the solution in solution explorer, and save as over the top of the .sln
6. In both Win32 and x64 configurations, get properties on the Box2D project
7. Change C++ Code Generation / Enable C++ Exceptions (Yes/EHsc)
8. In General / Output Directory - change to your new lib path and add a trailing slash.
9. Compile both Win32 and x64 configurations.

Build GLFW

1. Copy build VC13 to VC14
2. Create a lib/msvs2014/32 and 64 folders
3. Open solution is VS2015
4. Allow it to convert the project
5. Convert the solution by selecting the solution in solution explorer, and save as over the top of the .sln
6. In both Win32 and x64 configurations, get properties on the Box2D project
7. In General / Output Directory - change to your new lib path and add a trailing slash.
8. Compile both Win32 and x64 configurations.