Open CASCADE Technology

Guide for building third-party products on Windows

CONTENTS

1. INTRODUCTION

This document presents guidelines for building third-party products used by Open CASCADE Technology (OCCT) and samples on Windows platform.

In order to understand these guidelines, you need to be familiar with MS Visual Studio / Visual C++.

You need to use the same version of MS Visual Studio for building all third-party products and OCCT itself, in order to receive a consistent set of run-time binaries.

The links for downloading the third-party products are available on the web site of OPEN CASCADE SAS at http://www.opencascade.org/getocc/require/.
There are two types of third-party products which are used by OCCT:

1) Mandatory products: Tcl 8.5, Tk 8.5, TclX 8.4, FreeType 2.4.10, Ftgl 2.1.3
2) Optional products: TBB 3.0-018, gl2ps 1.3.5, FreeImage 3.14.1

It is recommended to create a separate new folder on your workstation where you will unpack the downloaded archives of the third-party products, and where you will build these products (for example, ***c:\occ3rdparty***).

Further in this document, this folder is referred to as ***<3rdparty>***.

2. BUILDING MANDATORY THIRD-PARTY PRODUCTS

2.1. Tcl/Tk 8.5

Tcl/Tk is required for DRAW test harness.

We recommend installing a binary distribution that could be downloaded from http://www.activestate.com/activetcl. Go to “Free Downloads“ and pick the version of the Install Wizard that matches your target platform – 32 bit (x86) or 64 bit (x64). The version of Visual Studio you use is irrelevant when choosing the Install Wizard.

Run the Install Wizard you downloaded, and install Tcl / Tk products to ***<3rdparty>\ tcltk-win32*** folder (for 32-bit platform) or to ***<3rdparty>\tcltk-win64*** folder (for 64-bit platform).

Further in this document, this folder is referred to as ***<tcltk>***.

2.2. FreeType 2.4.10

FreeType is required for Ftgl.

You can download its sources from http://sourceforge.net/projects/freetype/files/

The building process is the following:

1. Unpack the downloaded archive of FreeType 2.4.10 product into the ***<3rdparty>*** folder.

As a result, you should have a folder named ***<3rdparty>\freetype-2.4.10***. Rename it according to the rule: freetype-<platform>-<compiler>-<building mode>, where

<platform> is win32 or win64;

<compiler> is vc8 or vc9 or vc10;

<building mode> is opt (for release) or deb (for debug)

Further in this document, this folder is referred to as ***<freetype>***.

2. Open the solution file ***<freetype>\builds\"vc2005*** *or* ***vc2008 or vc2010"\\freetype.sln*** in Visual Studio.

3. Select a configuration to build.

Choose "**LIB Release Multithreaded**" if you are building Release binaries.
Choose "**LIB Debug Multithreaded**" if you are building Debug binaries.

4. Set Project->Properties->Configuration Properties->General->Configuration Type
Set this option to "**Dynamic Library (.dll)**".

Starting from OCCT 6.5.3 freetype library is used as a dynamic library

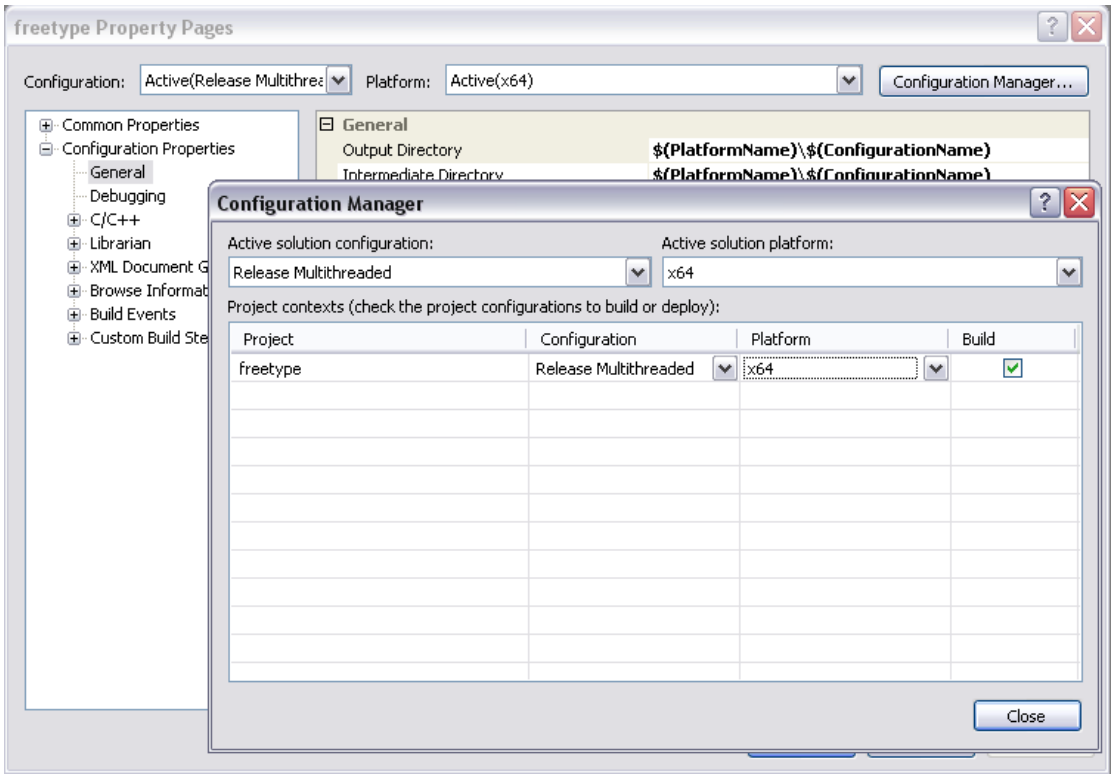
5. Set Project->Properties->Configuration Properties->C/C++->Advanced->Calling Convention
Set this option to **"__fastcall(/Gr)".**

6. Select a platform to build.

Choose "***Win32***" if you are building for 32 bit platform.

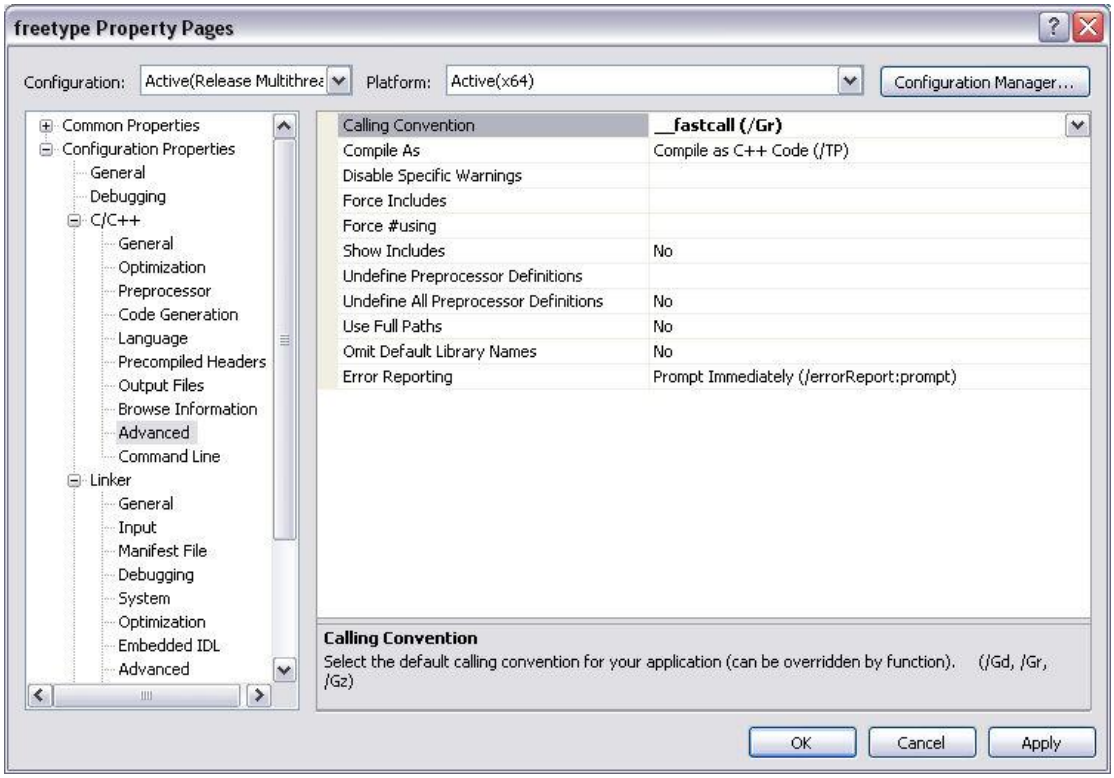
If you are building for 64 bit platform, then:

a) Using the Configuration Manager (Build -> Configuration Manager), add ***x64*** platform to the solution configuration chosen in step 3, by copying the settings from Win32 platform.



b) Set Project->Properties->Configuration Properties->C/C++->Advanced->Calling Convention.

Set this option to "***__fastcall(/Gr)***".



c) Finally, choose "***x64***" platform to build.

7. Start the building process of **freetype** project.

As a result, you should have the import library and dynamic library files named and located according to the parameters Project->Properties->Configuration Properties->Linker->Advanced->Import Library and Project->Properties->Configuration Properties->Linker->General->Output File respectively.

2.3. Ftgl 2.1.3

Ftgl is required for OCCT Visualization libraries.

This third-party product should be built as a dynamically loadable library (dll file). You can download its sources from http://sourceforge.net/projects/ftgl/files/
The building process is the following:

1. Unpack the downloaded archive of Ftgl 2.1.3 product (***ftgl-2.1.3.tar.gz***) into the ***<3rdparty>*** folder.

As a result, you should have a folder named ***<3rdparty>\FTGL***.

Rename it according to the rule: ftgl-<platform>-<compiler>-<building mode>, where

<platform> is win32 or win64;

<compiler> is vc8 or vc9 or vc10;

<building mode> is opt (for release) or deb (for debug)

Further in this document, this folder is referred to as ***<ftgl>***.

2. Open the workspace file ***<ftgl>\msvc\"vc71*** or ***vc8"\ftgl.sln*** in Visual Studio.

3. Select a configuration to build.

Choose "***Release MT***" if you are building Release binaries.

Choose "***Debug MT***" if you are building Debug binaries.

Note:

If you want to build a debug version of Ftgl binaries then you must rename the output file (Project->Properties->Configuration Properties->Linker->General->Output File) from ftgl _d.dll to ftgl.dll

Similarly, rename:

Project->Properties->Configuration Properties->Linker->Debugging->Generate Program Database File

from ftgl_d.pdb to ftgl_MTD.pdb

Project->Properties->Configuration Properties->Linker->Advanced->Import Library

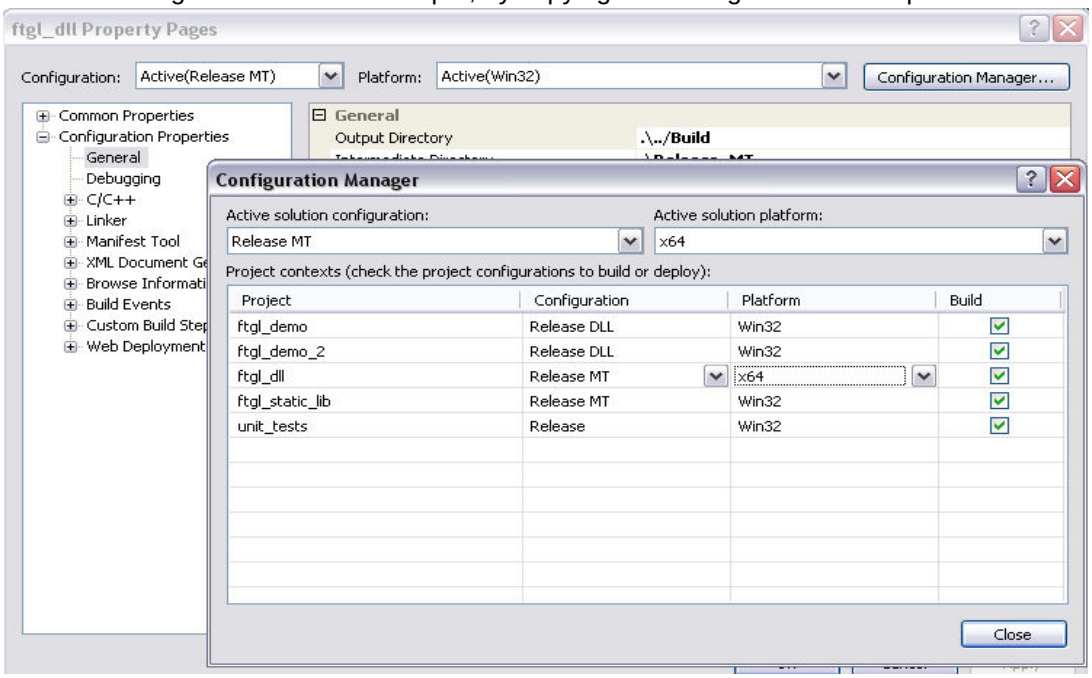
from ftgl_d.lib to ftgl.lib

4. Select a platform to build.

Choose "***Win32***" if you are building for a 32 bit platform.

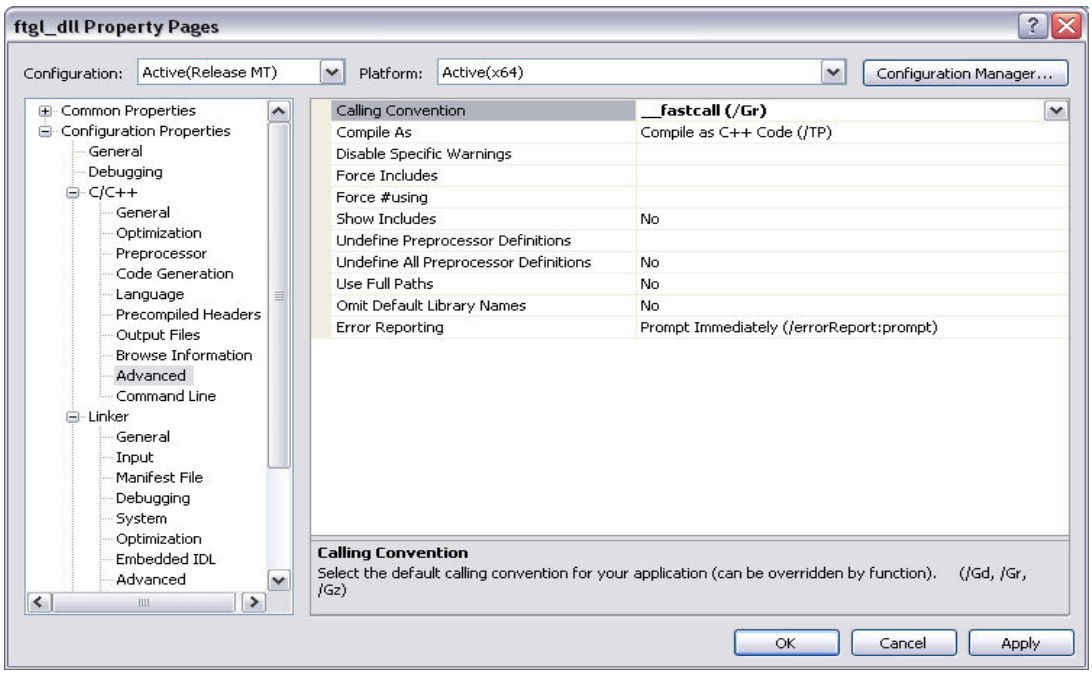
If you are building for a 64 bit platform:

a) Using the Configuration Manager (Build -> Configuration Manager), add ***x64*** platform to the solution configuration chosen in step 3, by copying the settings from Win32 platform.



b) Set Project->Properties->Configuration Properties->C/C++->Advanced->Calling Convention

Set this option to "***__fastcall(/Gr)***".



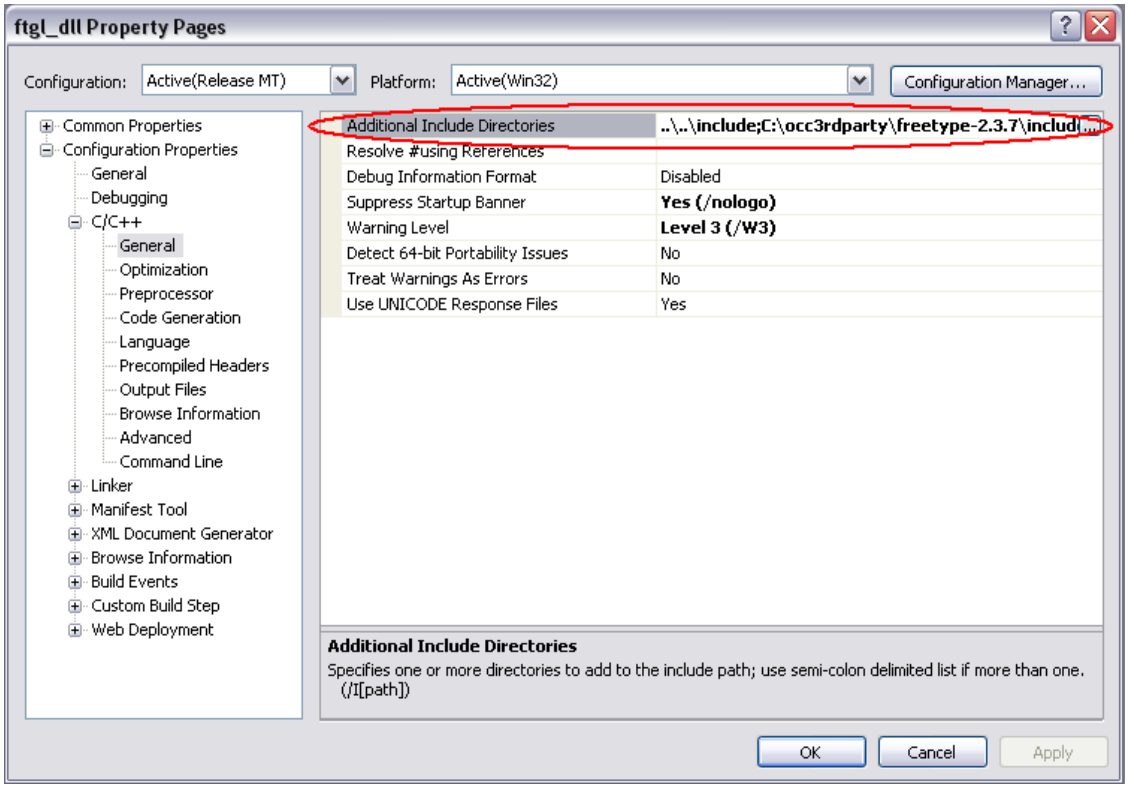
c) Finally, choose "***x64***" platform to build.

5. In Solution Explorer, set ***ftgl_dll*** as the StartUp project.

6. Set Project->Properties->Configuration Properties->C/C++->General->Additional Include Directories.

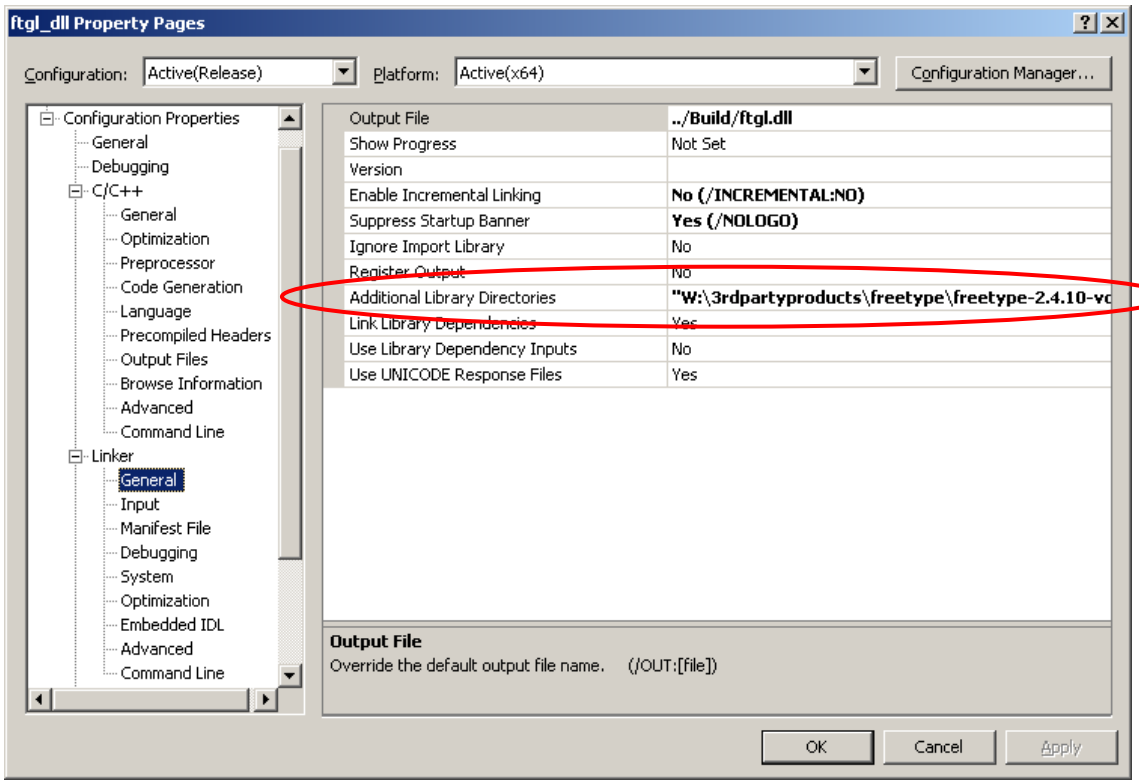
This list of paths should contain paths to header files of ftgl and freetype products, and should look like the following:

..\..\include;<freetype>\include



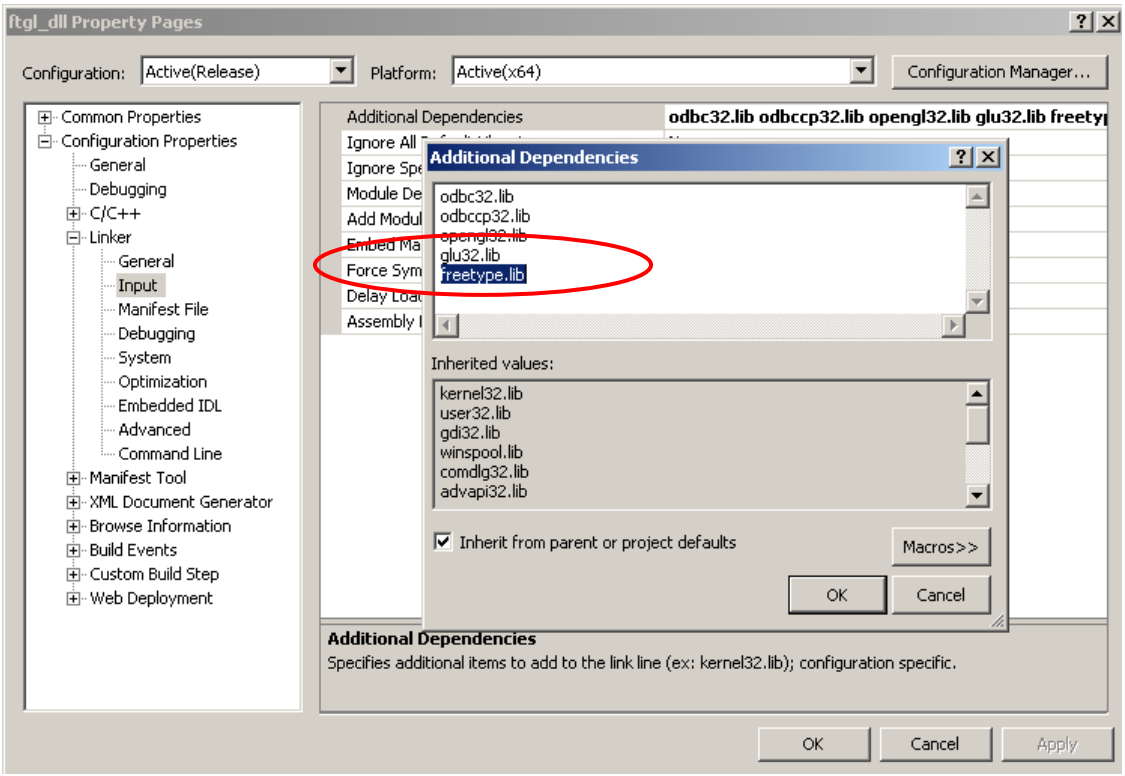
7. Set Project->Properties->Configuration Properties->Linker->General->Additional Library Directories.

This list of paths should contain path to the library file of freetype product built earlier, and should look like the following:



8. Set Project->Properties->Configuration Properties->Linker->Input-> Additional Dependencies.

Update the name of the import library file of freetype at the end of the list of additional libraries.



9. Start the building process of ftgl_dll project.

As a result, you should have the import library and dynamic library files named and located according to the parameters Project->Properties->Configuration Properties->Linker->Advanced->Import Library and Project->Properties->Configuration Properties->Linker->General->Output File respectively.

3. BUILDING OPTIONAL THIRD-PARTY PRODUCTS

3.1. TBB 3.0-018

This third-party product is installed with binaries from the archive that can be downloaded from http://threadingbuildingblocks.org/. Go to “Downloads / Commercial Aligned Release”, find the release version you need (tbb30_018oss) and pick the archive for Windows platform.

Unpack the downloaded archive of TBB 3.0 product (***tbb30_018oss_win.zip***) into the ***<3rdparty>*** folder.

As a result, you should have a folder named ***<3rdparty>\tbb30_018oss***. Further in this document, this folder is referred to as ***<tbb>***.

3.2. gl2ps 1.3.5

This third-party product should be built as a dynamically loadable library (dll file). You can download its sources from http://geuz.org/gl2ps/src/

The building process is the following:

1. Unpack the downloaded archive of gl2ps 1.3.5 product (***gl2ps-1.3.5.tgz***) into the ***<3rdparty>*** folder.

As a result, you should have a folder named ***<3rdparty>\gl2ps-1.3.5-source***.

Rename it according to the rule: gl2ps-<platform>-<compiler>-<building mode>, where

<platform> is win32 or win64;

<compiler> is vc8 or vc9 or vc10;

<building mode> - opt (for release) or deb (for debug)

Further in this document, this folder is referred to as ***<gl2ps>***.

2. Download (from http://www.cmake.org/cmake/resources/software.html) and install ***CMake*** build system.

3. Edit the file ***<gl2ps>\CMakeLists.txt***.

After line 113 in CMakeLists.txt:

set_target_properties(shared PROPERTIES COMPILE_FLAGS "-DGL2PSDLL -DGL2PSDLL_EXPORTS")

add the following line:

add_definitions(-D_USE_MATH_DEFINES)

4. Launch CMake (cmake-gui.exe) using the Program menu.

5. In CMake:

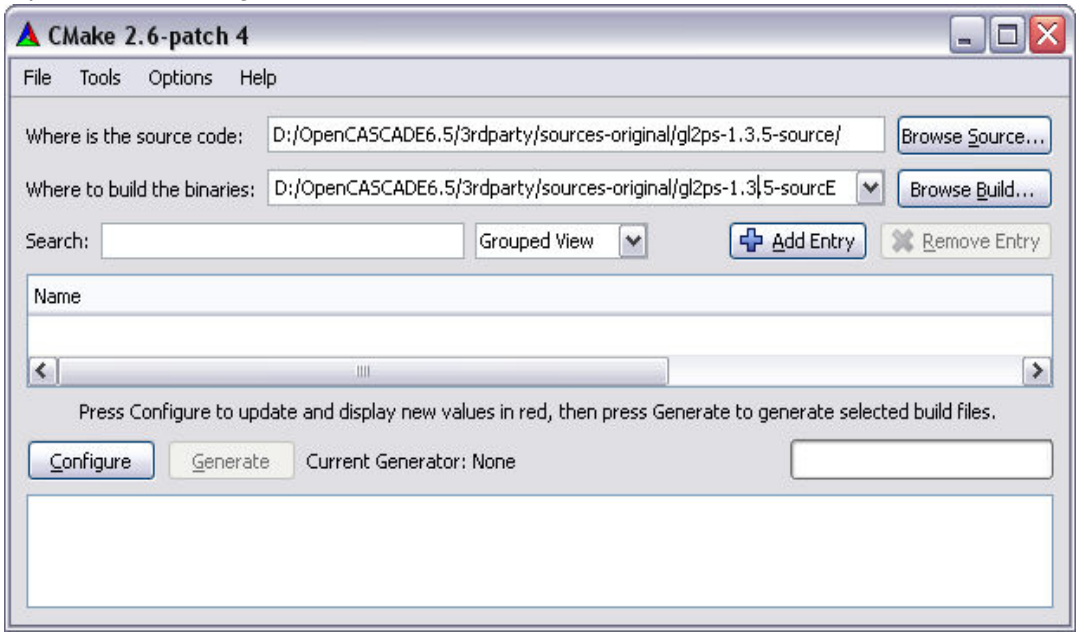
a) Define where the source code is.

This path must point to ***<gl2ps>*** folder.

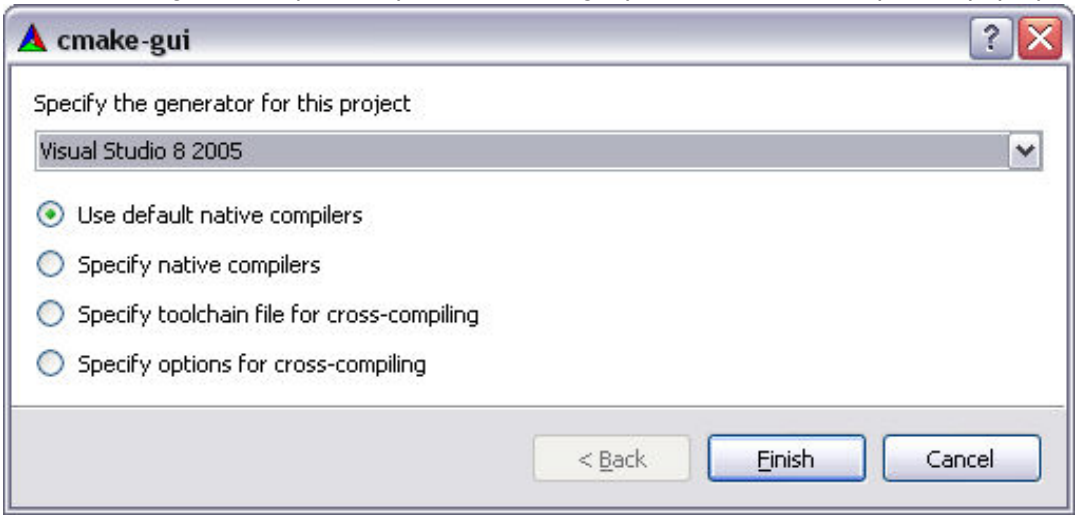
b) Define where to build the binaries.

This path must point to the folder where generated gl2ps project binaries will be placed (for example, ***<gl2ps>\bin***). Further in this document, this folder is referred to as ***<gl2ps_bin>***.

c) Press the “Configure” button.

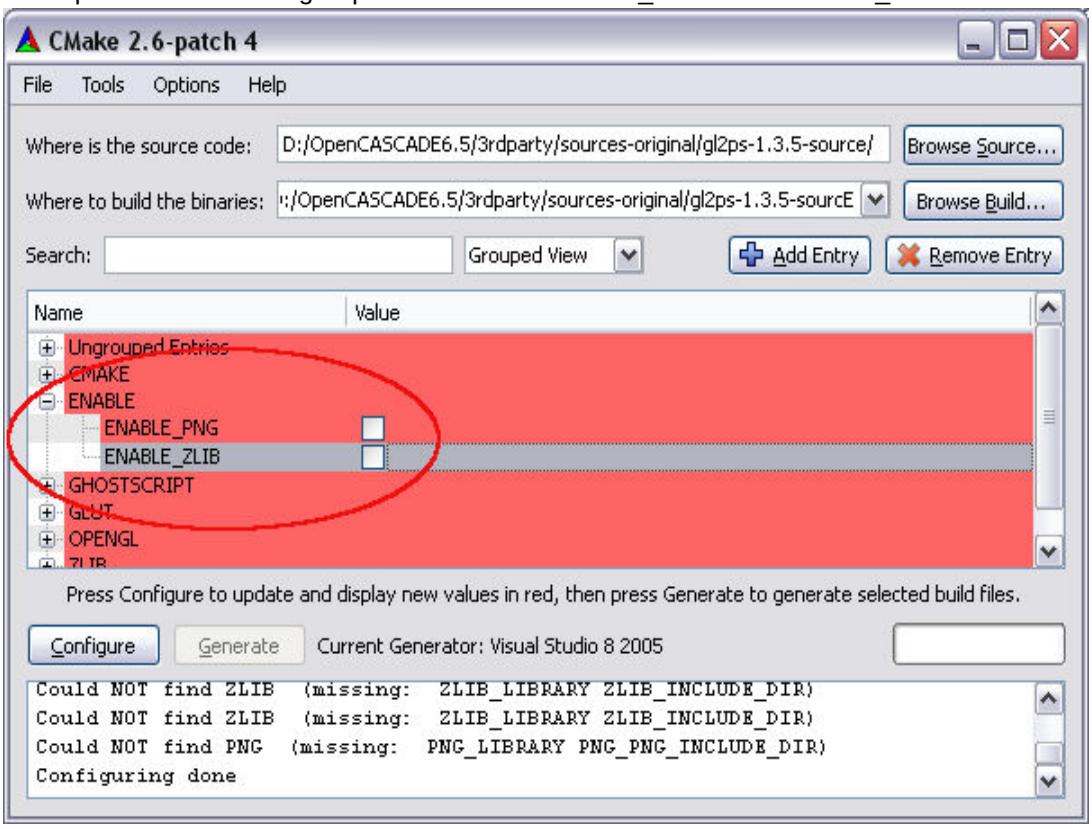


6. Select the generator (the compiler and the target platform - 32 or 64 bit) in the pop-up window.

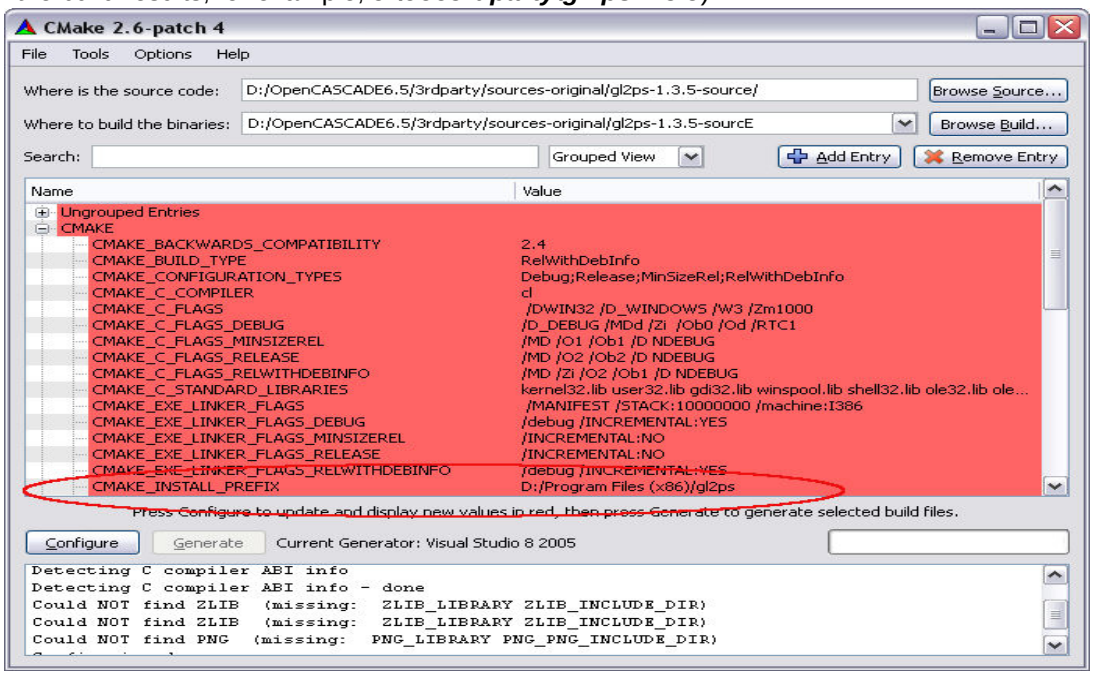


Then press the “Finish” button to return to the main CMake window.

6. Expand the ENABLE group and uncheck ENABLE_PNG and ENABLE_ZLIB check boxes.



7. Expand the CMAKE group and define CMAKE_INSTALL_PREFIX (path where you want to install the build results, for example, ***c:\occ3rdparty\gl2ps-1.3.5***).



8. Press the “Configure” button again, and then the “Generate” button in order to generate Visual Studio projects. After completion, close CMake application.

9. Open the solution file ***<gl2ps_bin>\gl2ps.sln*** in Visual Studio.

10. Select a configuration to build.

Choose "***Release***" if you are building Release binaries.

Choose "***Debug***" if you are building Debug binaries.

11. Select a platform to build.

Choose "***Win32***" if you are building for a 32 bit platform.

Choose "***x64***" if you are building for a 64 bit platform.

12. Build the solution.
13. Build the ***INSTALL*** project.

As a result, you should have the installed gl2ps product in the CMAKE_INSTALL_PREFIX path.

3.3. FreeImage 3.14.1

This third-party product should be built as a dynamically loadable library (.dll file). You can download its sources from http://sourceforge.net/projects/freeimage/files/Source%20Distribution/

The building process is the following:

1. Unpack the downloaded archive of FreeImage 3.14.1 product (***FreeImage3141.zip***) into ***<3rdparty>*** folder.

As a result, you should have a folder named ***<3rdparty>\FreeImage***.

Rename it according to the rule: freeimage-<platform>-<compiler>-<building mode>, where

<platform> is win32 or win64;

<compiler> is vc8 or vc9 or vc10;

<building mode> is opt (for release) or deb (for debug)

Further in this document, this folder is referred to as ***<freeimage>***.

2. Open the solution file ***<freeimage>\FreeImage.*.sln*** in Visual Studio. Choose the file that corresponds to the version of Visual Studio you use.

Since the version of Visual Studio you use is higher than 6, apply conversion of the workspace. Such conversion should be suggested automatically by Visual Studio.

If there is no solution file matching the version of Visual Studio you use, choose the solution file for the highest version of Visual Studio, then apply conversion. Such conversion should be suggested automatically by Visual Studio.

3. Select a configuration to build.

Choose "***Release***" if you are building Release binaries.

Choose "***Debug***" if you are building Debug binaries.

Note:

If you want to build a debug version of FreeImage binaries then you must rename the following files for projects FreeImage and FreeimagePlus:

Project->Properties->Configuration Properties->Linker->General->Output File

from FreeImage**d**.dll to FreeImage.dll

from FreeImagePlus**d**.dll to FreeImagePlus.dll

Project->Properties->Configuration Properties->Linker->Debugging->Generate Program Database File

from FreeImage**d**.pdb to FreeImage.pdb

from FreeImagePlus**d**.pdb to FreeImagePlus.pdb

Project->Properties->Configuration Properties->Linker->Advanced->Import Library

from FreeImage**d**.lib to FreeImage.lib

from FreeImagePlus**d**.lib to FreeImagePlus.lib

Project->Properties->Configuration Properties->Build Events->Post-Build Event->Comand Line

from FreeImage**d**.dll to FreeImage.dll

from FreeImage**d**.lib to FreeImage.lib

from FreeImagePlus**d**.dll to FreeImagePlus.dll

from FreeImagePlus**d**.lib to FreeImagePlus.lib

Additionally, for project FreeImagePlus rename:

Project->Properties->Configuration Properties->Linker->Input->Additional Dependencies

from FreeImage**d**.lib to FreeImage.lib

4. Select a platform to build.

Choose "***Win32***" if you are building for a 32 bit platform.

Choose "**x64**" if you are building for a 64 bit platform.

5. Start the building process.

As a result, you should have the library files of FreeImage product in the ***<freeimage>\Dist*** folder (FreeImage.dll and FreeImage.lib files) and in the ***<freeimage>\Wrapper\FreeImagePlus\dist*** folder (FreeImagePlus.dll and FreeImagePlus.lib files).

4. REFERENCES

[1] Open CASCADE Technology web site: http://www.opencascade.org