

Win32 环境编译 OpenSceneGraph 2.x

王锐

从 OpenSceneGraph 1.9.x 开始, OSG 开始使用 CMake 工具来完成源代码工程的生成工作; 而在以前的版本中 (OSG1.2 及以下版本), 用于编译源代码的 VC7 和 VC8 工程文件是直接附带的。CMake 是一个跨平台的编译工具, 可以自动测试操作平台的特性, 并使用简单的脚本语言来描述源代码工程的生成要求并生成对应平台环境的工程文件 (makefile), 其作用类似于 Linux 下的 autoconf。相比以往的源代码编译方式, 显然这种方式更加灵活, 用户也可以根据自己的需求自行添加删除组件。

在 Windows 环境下编译 OSG 2.x, 所需的前期准备有:

Visual Studio, MinGW 或者其他的 C++ 编程 IDE 平台。

CMake 工具, 建议使用 2.4.6 及以上的版本, 下载地址:

www.cmake.org

OpenSceneGraph-2.x 版源代码, 稳定版本的下载地址为:

www.openscenegraph.org/downloads/developer_releases/OpenSceneGraph-2.2.0.zip

最新开发版本的下载地址为:

http://www.openscenegraph.org/downloads/developer_releases/OpenSceneGraph-2.3.4.zip

也可以下载 SVN 版本, 使用命令 (前提是安装了 Subversion 软件):

`svn co http://www.openscenegraph.org/svn/osg/OpenSceneGraph/trunk OpenSceneGraph`

OSG 的第三方开发库支持, 可以自行去各个开源开发库的网站下载, 也可以从下面的地址下载整合好的开发库 Lib 文件和头文件:

openscenegraph.org/downloads/dependencies/3rdParty_Win32binaries_2005_05_10.zip

为了实现对各种格式的模型和图像文件的加载, OSG 需要大量的第三方开发库支持, 这其中主要包括:

FreeType 库	提供了对 ttf, ttc, cid, cff, cef, fon, fnt 格式的字体文件的加载支持。用户程序使用 osgdb_freetype.dll 插件实现对上述字体的加载。	OSG 2.2 及以下版本可以直接使用上面的整合下载地址下载 FreeType 2.19 的开发库文件。 OSG 2.3.x 版本增加了 Text3D 类, 因此需要下载较新版本的 FreeType 2.3.x: www.freetype.org
LibJPEG 库	提供对 jpg, jpe 格式的图片文件的加载支持, 用户程序使用 osgdb_jpeg.dll 实现加载。	可以直接下载整合的库文件。
LibPNG 库	提供对 png 格式的图片文件的加载支持, 用户程序使用 osgdb_png.dll 实现加载。	可以直接下载整合的库文件。
LibTiff 库	提供对 tif 格式的图片文件的加载支持, 用户程序使用 osgdb_tiff.dll 实现加载。	可以直接下载整合的库文件。
LibUnGIF 库	提供对 gif 格式的图片文件的加载支持, 用户程序使用 osgdb_gif.dll 实现加载。	可以直接下载整合的库文件。
ZLib 库	提供对 zip 格式的压缩文件的加载支持, 用户程序使用 osgdb_zip.dll 实现加载。	可以直接下载整合的库文件。

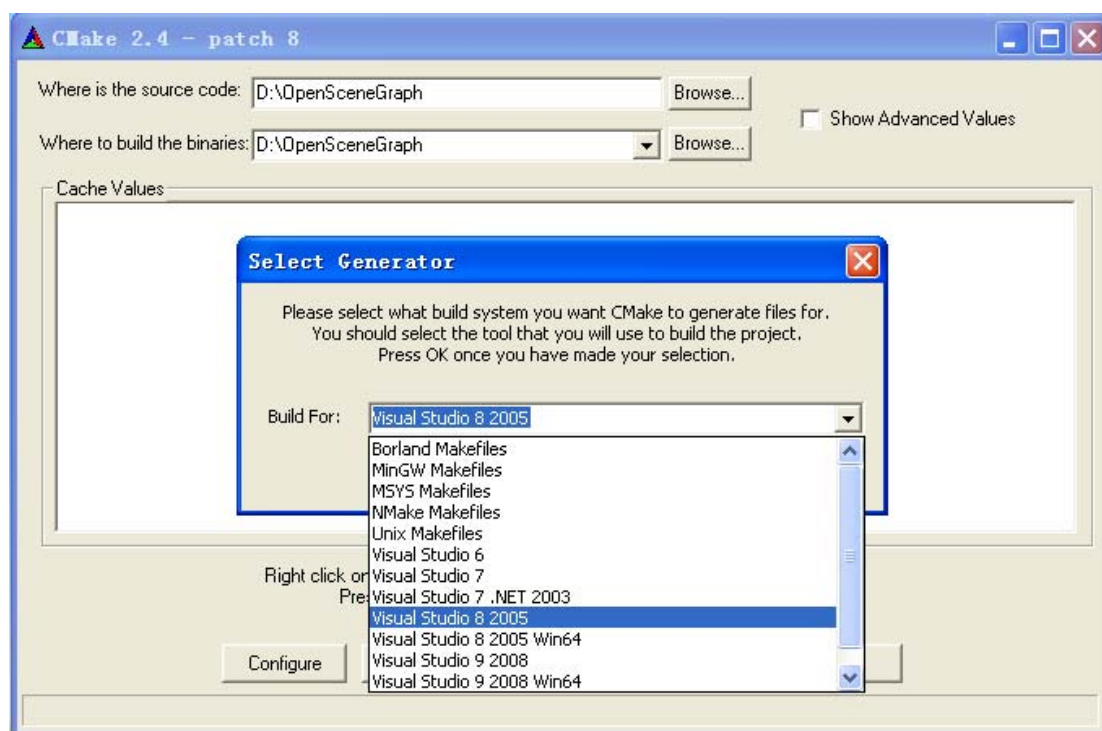
GLUT 库	OpenGL 应用工具包, GUI 库的一种。可以与 OSG 嵌合。	可以直接下载整合的库文件。
GDAL 库	GDAL 是专用于解析地理信息数据 (GIS) 文件的工具库, 并遵循 MIT/X 许可证开放源代码。	可以下载 GDAL 1.4 或以上版本: www.gdal.org
Collada 库	COLLADA 是免授权费用的开放标准技术, 使用基于 XML 的技术来描述 3D 内容, 其文件扩展名为 dae。用户程序使用 osgdb_dae.dll 实现加载。	需要下载相应的 Collada-Dom 库: www.collada.org
Inventor 库	Open Inventor 是 VRML 文件的官方格式, 由 SGI 公司制定其规范并开发, 其文件扩展名为 wrl。用户程序使用 osgdb_iv.dll 实现加载。	需要相应的 Open Inventor 库, 可以查询 SGI 网站获取其最新信息: oss.sgi.com/projects/inventor
Jasper 库	Jasper 由加拿大维多利亚大学的一位教师主持开发, 它支持“高压压缩、低比特速率”的 JPEG2000 图片格式, 其文件扩展名为 jp2。用户程序使用 osgdb_jp2.dll 实现加载。	需要下载相应的 Jasper 库: www.ece.uvic.ca/~mdadams/jasper
OpenVRML 库	OpenVRML 是一个可以为应用程序提供 VRML 支持的工具包, 其源代码还在不断完善中。文件扩展名为 wrl, 用户程序使用 osgdb_vrml.dll 实现加载。	所需的版本为 0.14.3, 更新的版本可能无法为 OSG 插件所用, 需要下载相应的 OpenVRML 库: www.openvrml.org
Performer 库	OpenGL Performer 是一个可扩展的高性能实时三维视景开发软件包, 有自己的数据格式类型, 其文件扩展名为 pfb, 用户程序使用 osgdb_pfb.dll 实现加载。	需要相应的 OpenGL Performer 库, 可以查询 SGI 网站获取其最新信息: oss.sgi.com/projects/performer/
QuickTime 库	QuickTime SDK 库提供了对 MOV 等媒体文件的强大支持, 它支持的文件扩展名包括 mov, avi, mpg, mpv, dv, mp4, m4v, psd, tga, jpg, jpe, jpeg, tif, tiff, gif, png 等, 用户程序使用 osgdb_qt.dll 实现加载。	需要下载相应的 QuickTime SDK 库, OSG 2.0 需要 QuickTime SDK 7.1.2 及以上版本的支持。可以登录 Apple 的网站并注册, 然后下载: developer.apple.com/quicktime
XINE 库	XINE 是一个 Linux 下的媒体播放开发库, 目前在 Windows 下尚无成熟的应用。它支持的文件扩展名包括 mov, mpg, mpv, dv, avi, wmv 等。	Windows 下可能无法编译。
FLTK 库	一种轻量级的 GUI 库。可以与 OSG 嵌合。	可以下载 FLTK 1.1 或以上版本: www.fltk.org
Qt 库	最为强大的跨平台 GUI 库之一。可以与 OSG 嵌合。	只能下载开源版本, Qt3/4 的版本都是可以的: trolltech.com/products/qt
Fox 库	一款性能出色的 GUI 库。可以与 OSG 嵌合。	可以下载 FOX 1.6 或以上版本: www.fox-toolkit.org
wxWidgets 库	强大的跨平台 GUI 库之一, 其代码特点	可以下载 2.8.x 或以上版本:

	与 MFC 类似，遵循 LGPL 协议。可以与 OSG 嵌合。	www.wxwidgets.org
SDL 库	一款历史悠久的 GUI 库，图形和声音的处理能力较强，遵循 LGPL 协议。可以与 OSG 嵌合。	可以下载 1.2 或以上版本： www.libsdl.org

准备好所需的源代码，编译工具和第三方开发工具后，就可以开始在 Windows 环境下编译 OSG 2.x 了。

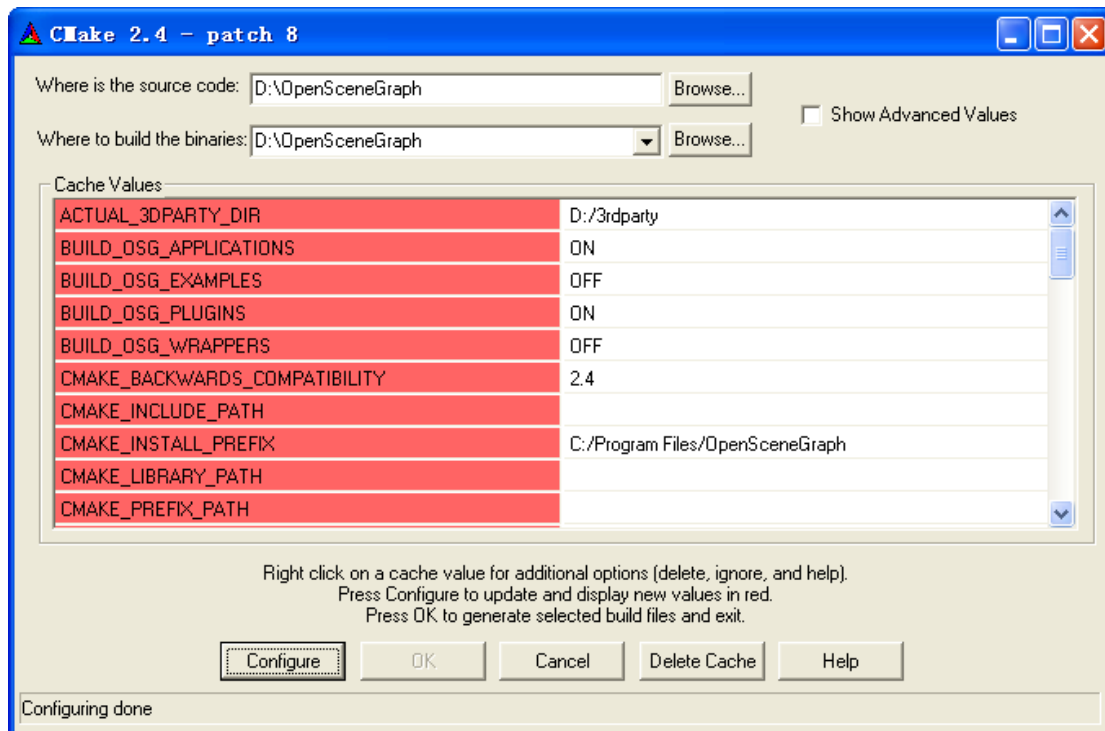
1、打开 CMake 的界面，将 OpenSceneGraph-2.x.zip 解压缩，并将解压目录中的 CMakeLists.txt 文件拖动到 CMake 的界面下。

2、在弹出的对话框中，选择与当前平台所对应的编程环境，一般来说这是自动选择的，用户也可以根据自己的需求生成其它编程环境下的 Makefile 或者工程文件。



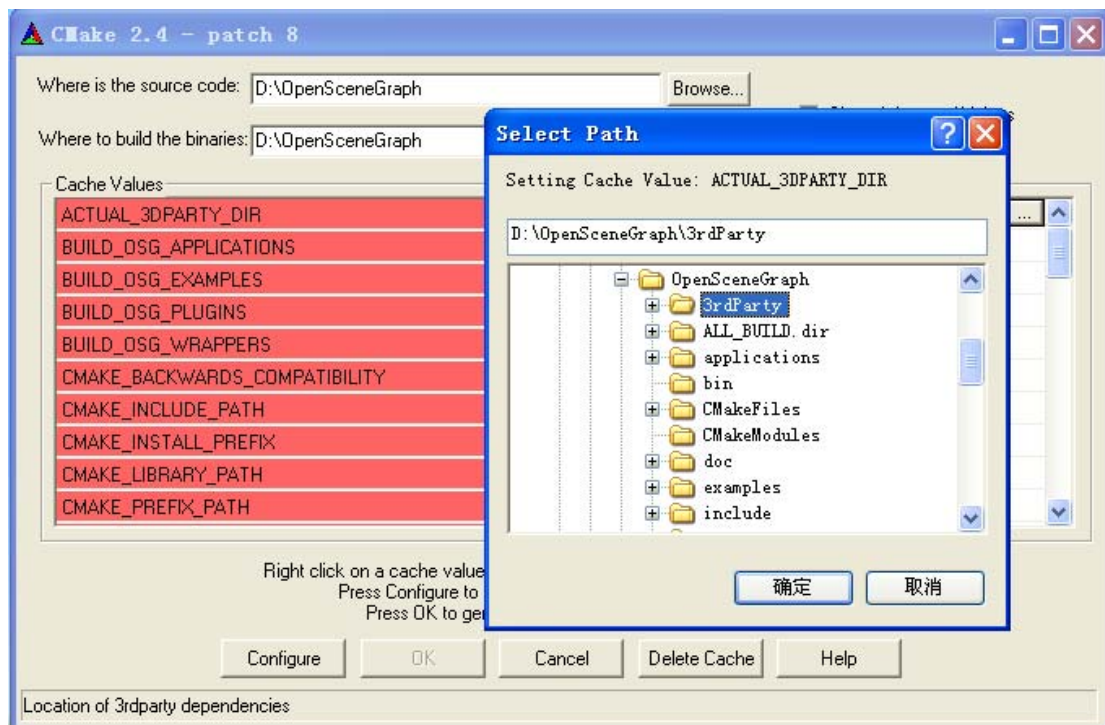
3、按下 **Configure** 按钮，此时列表框中出现一系列可以修改的参数值，在完成所有的修改之前，标识配置完成的“OK”按钮不会被启用，因此也不会生成新的工程文件。图中红色的选项说明该项需要立即进行配置；灰色的选项说明该项已经配置完毕，不过也可以重新进行配置。按下“Configure”按钮进行本次配置，而与当前所配置的项相关联的选项将在下一次配置时再次变红，等待用户重新进行配置；全部的选项都变成灰色后，“OK”按钮可以被按下，此时将生成用户所需的所有工程文件。

“Configure”按钮被按下多少次都没有关系，系统会自动判断用户是否更改了选项参数，并据此列出新的关联选项；如果用户没有修改，那么再次按下“Configure”后所有选项都呈现灰色，直到用户修改其中某项的参数或者按下“OK”为止。



在第一次进行配置时，有以下几项需要注意：

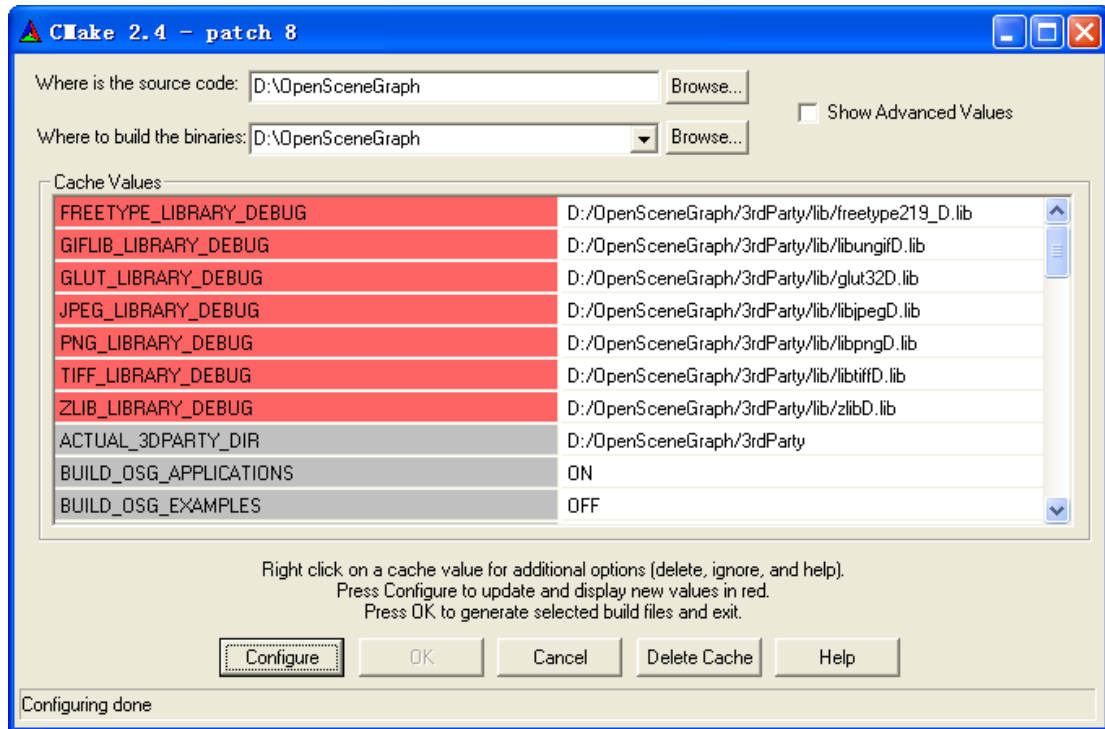
ACTUAL_3DPARTY_DIR：即整合的第三方开发库所在目录，如果下载了 OSG 提供的整合好的第三方开发库，那么可以在这里设置解压之后它的位置，并直接按下 **Configure** 按钮。CMake 将在下次配置时自动搜索所需的 Lib 文件和头文件路径。



CMAKE_CONFIGURATION_TYPES：定义了工程文件中工程配置的种类，缺省值包括 Debug, Release, MinSizeRel 和 RelWithDebInfo 四种类型。注意不同的配置类型在工程中对应不同的编译选项，可以在 **CMAKE_CXX_FLAGS** 和 **CMAKE_EXE_LINKER_FLAGS** 中进行修改。

CMAKE_INSTALL_PREFIX: 用于以后保存编译生成的 EXE, DLL 和 LIB 文件的路径。

4、按下“Configure”，进入第二次配置，如图所示：



可以看到，FreeType，LibUnGIF 等第三方库的位置已经被正确找到，在此后生成的工程文件中，这些库文件的引用和参考目录将被自动添加。

此外还需要注意的是：

BUILD_OSG_EXAMPLES: 是否编译 OSG 所有例子程序的选项，应当选择“ON”。

BUILD_OSG_WRAPPERS: 是否编译 OSG 内省/反射库，通常我们也可以设置为“ON”。

BUILD_MFC_EXAMPLE: 是否编译 MFC 与 OSG 嵌合的例子，这一选项只有在选择编译 OSG 的例子程序，并再次按下“Configure”之后才会出现。通常我们选择“ON”。

此时如果再次按下“Configure”，确认上述的修改，那么“OK”按钮将可以使用；按下“OK”按钮后即可生成所有的工程文件，但是可能存在一些问题，例如：OSG 的 FreeType 插件可能无法被正确地编译出来，还有其他一些基于第三方开发库的插件，如 osgdb_qt 等，可能无法生成。

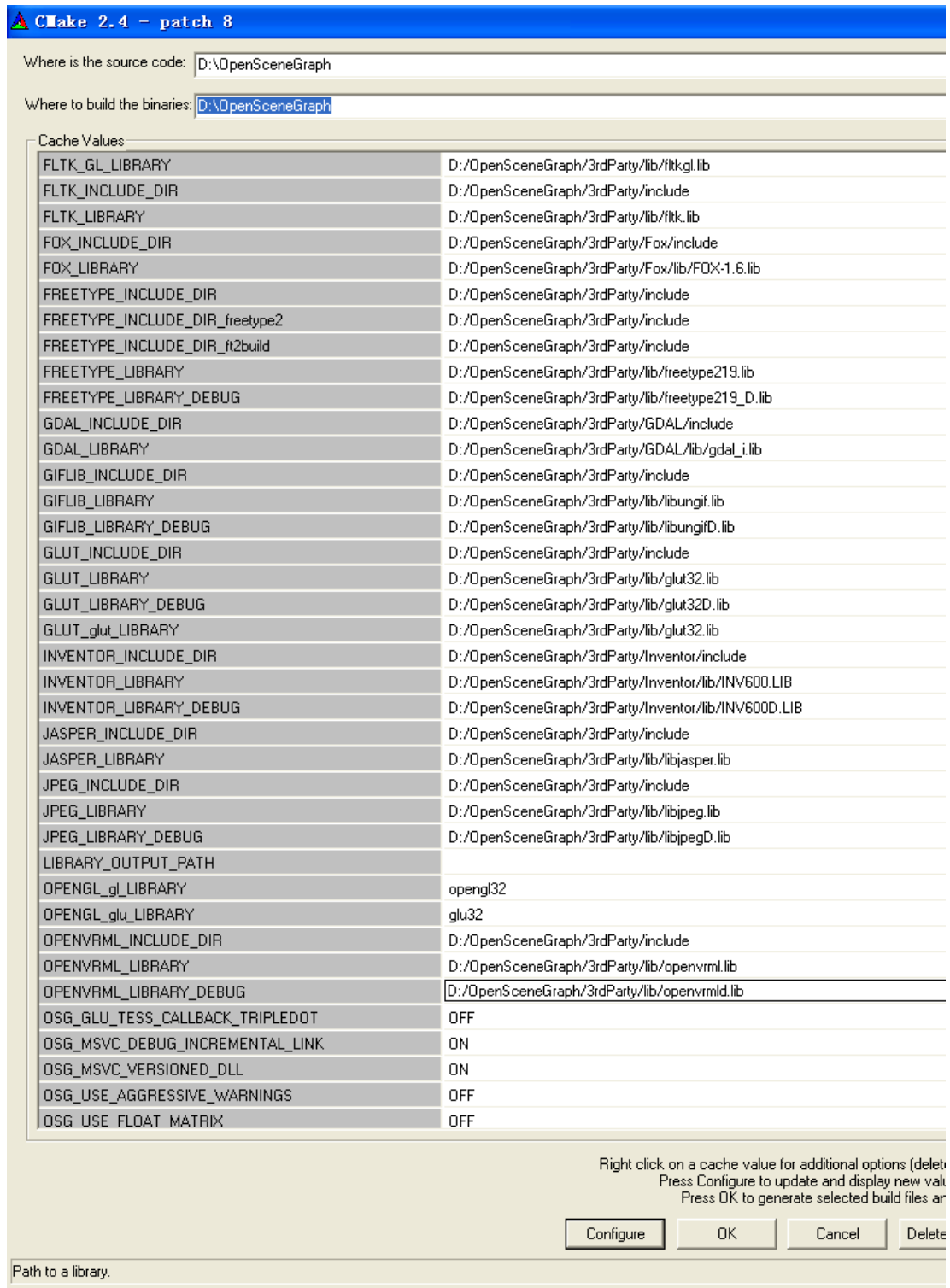
5、先不要急于按下“Configure”，点击右上的选框“Show Advanced Values”，可以看到列表框中的选项增加了。这其中很多选项是之前未经选择的。这其中需要重新进行配置的主要是各个第三方开发库的头文件路径和 Lib 文件路径；如果用户平台上还有其他 GUI 库，例如 Qt，Fltk，wxWidgets 等，那么也需要在这里重新进行配置。

配置第三方开发库的相关参数时，主要需要配置以下两个参数的值：

……_INCLUDE_DIR: 头文件的位置。

……_LIBRARY: 需要链接的静态库文件。

……_LIBRARY_DEBUG: 相应的 DEBUG 库文件。



由于 CMake 并不能自动判断所有的第三方链接库的配置情况，因此更多时候用户需要自己动手进行这两项的配置，配置的具体方法如下表，其中略去了绝对路径，且使用头文件的名称来表示 INCLUDE_DIR（事实上应当用绝对路径的目录名）。

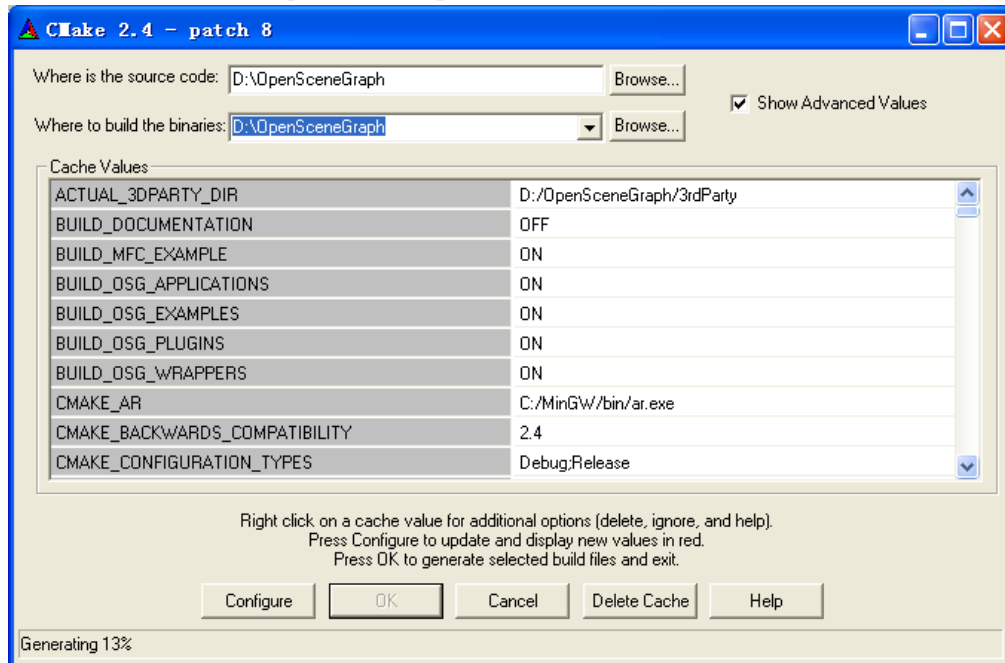
例如，ft2build.h，则 INCLUDE_DIR 应当为这个头文件所在的那一级子目录；FL/Fl.h，则 INCLUDE_DIR 应当为 FL 目录的上一级子目录。

下表是笔者编译 OSG 2.2 及 OSG 2.3.x 时使用的 Cmake 配置参数，由于所使用的第三方依赖库的差异，不同的人在编译时可能会有所差异，请根据实际情况自行调整。

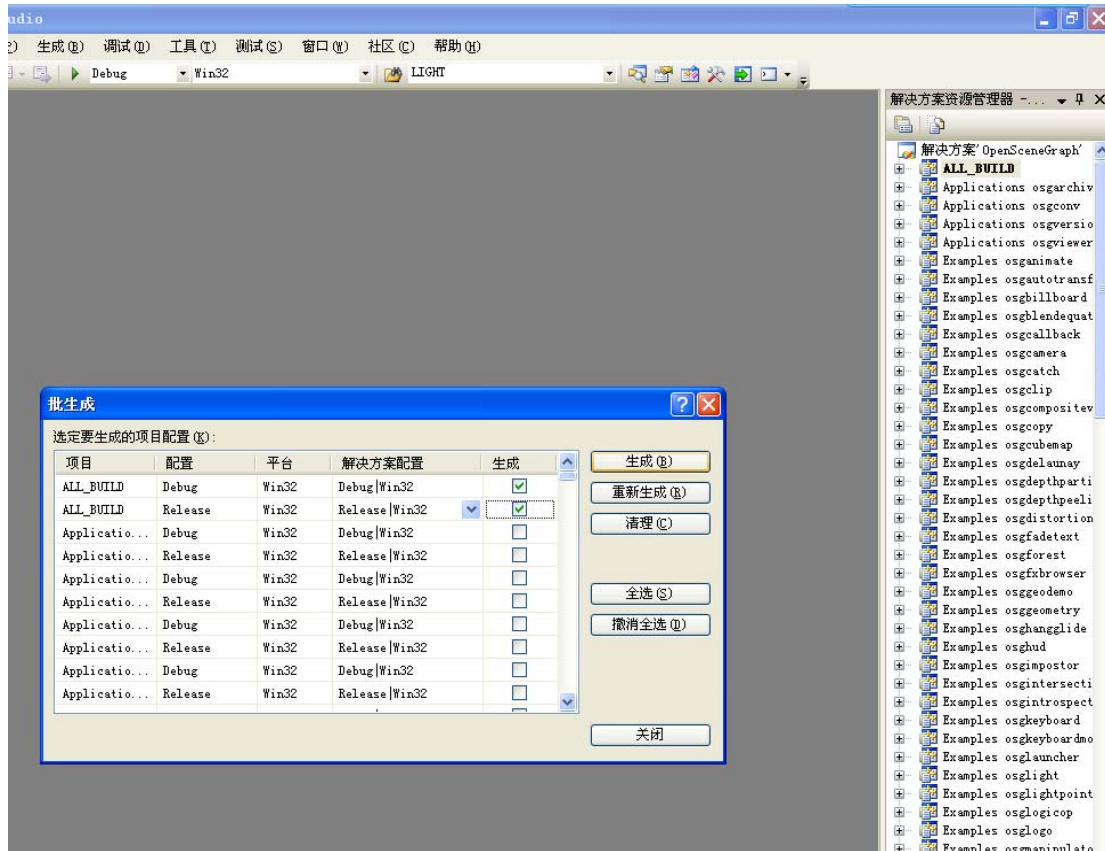
COLLADA	COLLADA_INCLUDE_DIR: dae.h COLLADA_LIBRARY: libcollada141dom13.lib
FLTK	FLTK_INCLUDE_DIR: Fl/Fl.h FLTK_LIBRARY: fltk.lib FLTK_GL_LIBRARY: fltkgl.lib
FOX	FOX_INCLUDE_DIR: fx.h FOX_LIBRARY: FOX-1.6.lib
FREETYPE	FREETYPE_INCLUDE_DIR_ft2build: ft2build.h FREETYPE_INCLUDE_DIR_freetype2: freetype/config/ftheader.h FREETYPE_LIBRARY: freetype235.lib
GDAL	GDAL_INCLUDE_DIR: gdal.h GDAL_LIBRARY: gdal_i.lib
GIFLIB	GIFLIB_INCLUDE_DIR: gif_lib.h GIFLIB_LIBRARY: libungif.lib
GLUT	GLUT_INCLUDE_DIR: GL/glut.h GLUT_LIBRARY: glut32.lib GLUT_glut_LIBRARY: glut.lib
INVENTOR	INVENTOR_INCLUDE_DIR: Inventor/So.h INVENTOR_LIBRARY: INV600.LIB
JASPER	JASPER_INCLUDE_DIR: jasper/jasper.h JASPER_LIBRARY: jasper.lib
JPEG	JPEG_INCLUDE_DIR: jpeglib.h JPEG_LIBRARY: libjpeg.lib
OPENVRML	OPENVRML_INCLUDE_DIR: openvrml/common.h OPENVRML_LIBRARY: openvrml.lib
PERFORMER	PERFORMER_INCLUDE_DIR: Performer/pfdu.h PERFORMER_LIBRARY: libpf.lib
PNG	PNG_INCLUDE_DIR: png.h PNG_PNG_INCLUDE_DIR: png.h PNG_LIBRARY: libpng.lib
QT	QT_……_INCLUDE_DIR: QT 的头文件目录 QT_……_LIBRARY: ……所指的 LIB 文件所在位置 QT_……_EXECUTABLE: ……所指的 EXE 文件所在位置
QUICKTIME	QUICKTIME_INCLUDE_DIR: QuickTime.h QUICKTIME_LIBRARY: qtmlClient.lib
SDL	SDL_INCLUDE_DIR: SDL.h SDLMAIN_LIBRARY: sdlmain.lib SDL_LIBRARY_TEMP: sdl.lib
TIFF	TIFF_INCLUDE_DIR: tiff.h TIFF_LIBRARY: libtiff.lib
XINE	在 Windows 下可能无法编译
ZLIB	ZLIB_INCLUDE_DIR: zlib.h ZLIB_LIBRARY: zlib.lib
wxWidgets	wxWidgets_ROOT_DIR: wxWidget 的安装目录

wxWidgets_LIB_DIR: wxWidgets 的 LIB 文件所在目录

6、确认所有的配置选项都变成灰色后，按下“OK”。对于 VC7, VC8 系列，此时将生成新的解决方案文件 OpenSceneGraph.sln。



打开.sln 工程文件，选择“批生成”，并选择编译生成 ALL_BUILD，即整个 OpenSceneGraph 工程。

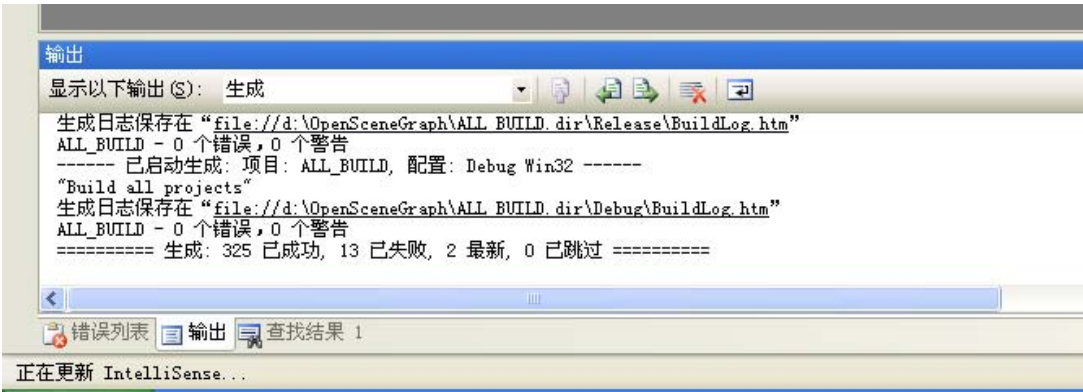


对于 VC6, CMake 也可以生成相应的 dsw 工程文件，但是编译时要注意标准 C++ STL

模板与 VC6 模板的冲突问题，可能需要选择安装 STLPort 支持，而且 OSG 官方已经不再提供针对 VC6 的技术支持。

对于 VC8，可能存在源代码中某些字符无法被识别的问题，此时该字符将显示为“?”，而直接在 VC 中进行编译将无法通过，需要手动将源代码中未被识别的字符改回原样。（osgversion 工程和 QuickTime SDK 的头文件中可能会出现此错误，将“?”改为下引号即可）

7、一些需要第三方支持的插件编译过程中，难免会出现编译错误，或者“LNK2019”的错误，这是由于 VC 的编译器不规范或者某些必要的链接库没有被引用所致，此时可以注意观察错误的详细提示信息，修改代码或添加必要的 Lib 库。链接时也可能出现函数冲突的错误，根据需要设置“/NODEFAULTLIB:xxx.lib”即可。



下面列出一些笔者编译时遇到的错误：

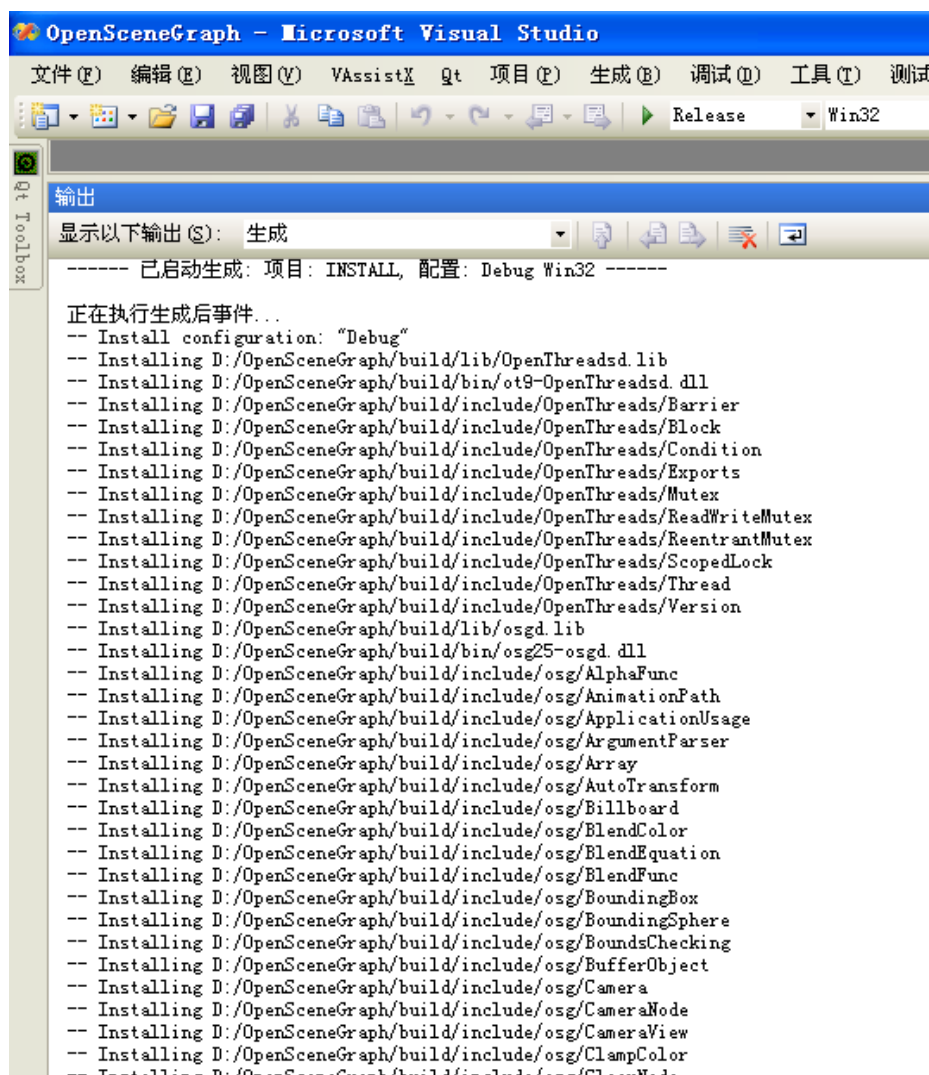
FLTK	编译通过，链接时出现 LNK2019 的错误。	工程的链接库一栏中缺少 comctl32.lib 和 wsock32.lib，添加即可。
Jasper	编译通过，链接时出现库的冲突。	选择忽略库 libcmtd.lib。
Inventor	编译不通过，出现类型未定义的错误。	部分类型在 Win32 环境中没有定义。在出错的文件中添加红色的行： #include <assert.h> typedef signed char int8_t; typedef short int16_t; typedef unsigned short uint16_t;
OpenVRML	OpenVRML 本身无法正确编译。	下载 0.14.3 的地址： sourceforge.net/project/downloading.php?groupname=artoolkit&filename=OpenVRML-0.14.3-win32.zip 此外，需要对 OpenVRML 的部分源文件进行修改，主要是旧有的 C 函数格式与 C++标准的冲突问题，简单进行修改后即可编译通过。
wxWidgets	出现与 wxGLCanvas 相关的 LNK2019 错误。	这不是编译时 OSG 的问题，而是我们所使用的 wxWidgets 没有开启 OpenGL 功能项。重新设置 lib 目录下 setup.h 文件中 wxUSE_GLCANVAS 参数为 1，并再次编译 wxWidgets。
Qt	出现 QmultiMap 字样	这一问题只出现在使用 VS8 SP1 的用户中，这是

	的编译错误。	<p>VS 编译器的一个新问题，需要修改 Qt 的 QMap.h 源代码：</p> <p>添加红色的行：</p> <pre>template <class Key, class T> class QMultiMap : public QMap<Key, T> { public: typedef typename QMap<Key, T>::iterator QMapIteratorType;</pre> <p>...</p> <p>然后修改两处（第二处的函数为 insert 而非 replace）：</p> <pre>Q_INLINE_TEMPLATE Q_TYPENAME QMap<Key, T>::iterator QMultiMap<Key, T>::replace(const Key &akey, const T &avalue) 为 Q_INLINE_TEMPLATE Q_TYPENAME QMultiMap<Key, T>::QMapIteratorType QMultiMap<Key, T>::replace(const Key &akey, const T &avalue)</pre>
Wrapper osgSim	出现 LNK2019 错误，且全部与目标文件 OpenFlightSimulator.obj 相关。	<p>从 OSG 2.2 升级到 OSG SVN 版本时，这个问题困扰了笔者很久，但是其实原因很简单：</p> <p>OSG 2.3.x 中去除了 osgSim:: OpenFlightSimulator 类以及相应的文件，但是 OSG 2.2 中该文件还存在，因此直接使用 SVN 版本覆盖源文件时不会覆盖该文件，造成工程链接时无法识别该文件的信息。删除源文件即可。</p>

8、所有的库文件，插件和示例程序都编译生成之后，就可以将它们安装到之前通过 CMAKE_INSTALL_PREFIX 参数指定的路径中了。打开“批生成”界面并选择生成 INSTALL 工程。



可以看到系统正在执行文件的安装工作：



```
----- 已启动生成: 项目: INSTALL, 配置: Debug Win32 -----

正在执行生成后事件...
-- Install configuration: "Debug"
-- Installing D:/OpenSceneGraph/build/lib/OpenThreads.lib
-- Installing D:/OpenSceneGraph/build/bin/ot9-OpenThreads.dll
-- Installing D:/OpenSceneGraph/build/include/OpenThreads/Barrier
-- Installing D:/OpenSceneGraph/build/include/OpenThreads/Block
-- Installing D:/OpenSceneGraph/build/include/OpenThreads/Condition
-- Installing D:/OpenSceneGraph/build/include/OpenThreads/Exports
-- Installing D:/OpenSceneGraph/build/include/OpenThreads/Mutex
-- Installing D:/OpenSceneGraph/build/include/OpenThreads/ReadWriteMutex
-- Installing D:/OpenSceneGraph/build/include/OpenThreads/ReentrantMutex
-- Installing D:/OpenSceneGraph/build/include/OpenThreads/ScopedLock
-- Installing D:/OpenSceneGraph/build/include/OpenThreads/Thread
-- Installing D:/OpenSceneGraph/build/include/OpenThreads/Version
-- Installing D:/OpenSceneGraph/build/lib/osgd.lib
-- Installing D:/OpenSceneGraph/build/bin/osg25-osgd.dll
-- Installing D:/OpenSceneGraph/build/include/osg/AlphaFunc
-- Installing D:/OpenSceneGraph/build/include/osg/AnimationPath
-- Installing D:/OpenSceneGraph/build/include/osg/ApplicationUsage
-- Installing D:/OpenSceneGraph/build/include/osg/ArgumentParser
-- Installing D:/OpenSceneGraph/build/include/osg/Array
-- Installing D:/OpenSceneGraph/build/include/osg/AutoTransform
-- Installing D:/OpenSceneGraph/build/include/osg/Billboard
-- Installing D:/OpenSceneGraph/build/include/osg/BlendColor
-- Installing D:/OpenSceneGraph/build/include/osg/BlendEquation
-- Installing D:/OpenSceneGraph/build/include/osg/BlendFunc
-- Installing D:/OpenSceneGraph/build/include/osg/BoundingBox
-- Installing D:/OpenSceneGraph/build/include/osg/BoundingSphere
-- Installing D:/OpenSceneGraph/build/include/osg/BoundsChecking
-- Installing D:/OpenSceneGraph/build/include/osg/BufferObject
-- Installing D:/OpenSceneGraph/build/include/osg/Camera
-- Installing D:/OpenSceneGraph/build/include/osg/CameraNode
-- Installing D:/OpenSceneGraph/build/include/osg/CameraView
-- Installing D:/OpenSceneGraph/build/include/osg/ClampColor
-- Installing D:/OpenSceneGraph/build/include/osg/ClampDepth
```

在这个路径下会自动建立 bin, lib, include 和 share 目录, 以便区分存放 DLL, LIB, 头文件和可执行的示例程序。

最后, 我们可以设定常用的系统环境变量, 并开始我们的 OpenSceneGraph 之旅了。