Building the Spout libraries

The Spout SDK can be built as a dll or static library which may be preferable to using source code. The same applies for a DirectX versions “SpoutDX”, SpoutDX12”, “SpoutDX9”.
“SpoutLibrary” is a C-compatible dll that can be used with other compilers.
All libraries can be built using Cmake. (https://cmake.org).

An example of using a dll or static library rather than source code can be found in :
“..\SPOUTSDK\SpoutDirectX\SpoutDX\Tutorial04_Lib\”
The libraries have to be created before the example can be built. Details can be found in “readme.md”.

CMAKE build

Installation

1) Download and install CMake for Windows. The easiest way is to use the installer. At the time of writing this was "cmake-3.25.1-x86-64.msi". Get it from the downloads page (https://cmake.org/download/).
2) Run the installer. Default options are OK, but it's useful to create a Desktop icon. "Finish" to complete.

Generating a project

1) On the desktop, find the CMake icon and open the CMake GUI.
2) For "Where is the source code:", click "Browse Source", navigate to wherever you saved the Spout repository and select the root folder, which could be either "Spout2-master" or “Spout2-beta”.
3) For "Where to build the binaries:", click "Browse Build" and navigate to "Spout2-master\BUILD".
4) At bottom click "Configure" to open the configuration dialog.
5) For "Specify the generator for this project", select your compiler and other options you may require. "Optional platform ..." will be empty. Default build is 64 bit. Leave it at that for now and other defaults. Click "Finish".

After completion you will see various build settings in red.

- ✔ SKIP_INSTALL_ALL – do not generate an INSTALL project to produce header and library folders. Default is ON. The INSTALL project is a separate build. Default is off for the following two options.
- ✔ SKIP_INSTALL_HEADERS – do not generate header files with the INSTALL project.
- ✔ SKIP_INSTALL_LIBRARIES – do not generate library files with the INSTALL project.
- ✔ SPOUT_BUILD_ARM – build for Windows on Arm for Arm architecture processors.
- ✔ SPOUT_BUILD_CMT - for Visual Studio compilers, this sets a project option “C/C++ > Code Generation > Runtime Library > Multi-threaded (/MT)” to compile the Visula Studio runtime libraries into the dll. Then the user does not need to install them separately. Check it off if you require compatibility with other libraries built “/MD”.
- ✔ SPOUT_BUILD_LIBRARY - builds a C-compatible library "SpoutLibrary" which could be of interest if you are not using Visual Studio.
- ✔ SPOUT_BUILD_SPOUTDX - builds the Spout DirectX11 support class "SpoutDX" as a dynamic link library. Default is off.

Finally Click "Generate".

Building the projects

When you see "Generating done", click "Open Project" within CMake. To open the project subsequently, for Visual Studio, the project is “Spout.sln” in the “BUILD” folder. In the compiler IDE you will see some or all of the following projects :

ALL_BUILD	
INSTALL	
Spout	(Spout SDK dll)
Spout_static	(Spout SDK static library)
SpoutDX	(Spout for DirectX 11 – option)
SpoutDX_static	(SpoutDX static library)
SpoutDX12	(Spout for DirectX 12 – option)
SpoutDX12_static	(SpoutDX12 static library)
SpoutDX9	(Spout for DirectX 9 – option)
SpoutDX9_static	(SpoutDX9 static library)
SpoutLibrary	(C compatible Spout library - option)
ZERO_CHECK	

ALL_BUILD

ALL_BUILD is the default “Startup Project”. Change to “Release” configuration and “Build > Build Solution”. When it has finished, browse to the "BUILD" folder and in the "Binaries" sub-folder you will find :

x64
- Spout.dll
- Spout.lib
- Spout_static.lib
- SpoutDX.dll
- SpoutDX.lib
- SpoutDX_static.dll
- SpoutDX9.dll
- SpoutDX9.lib
- SpoutDX9_static.dll
- SpoutDX12.dll
- SpoutDX12.lib
- SpoutDX12_static.dll
- SpoutLibrary.dll
- SpoutLibrary.lib

DirectX examples

DirectX examples Tutorial04 and Tutorial07 can be built using Cmake individually using the respective CmakeLists.txt file for each project. The “build” folder should be selected as the detination.

INSTALL

This is a separate project that produces all the files you need for the libraries in conveniently arranged folders :

```
bin      - dll files
lib      - library files
include - header diles
              SpoutDX
              SpoutDX9
              SpoutDX12
              SpoutGL
              SpoutLibrary
```

ZERO_CHECK

ZERO_CHECK can be run to find if any of the CMake files have changed since the last build and is not normally necessary.

Changing the CMake options

1) Close compiler IDE
2) Start CMake GUI again if it has been closed
3) Select any of the options available and check ON or OFF
4) Click "Generate" again to set the new options.
5) "Open Project" and Rebuild

Changing Generator or Platform

1) From the CMake GUI select "File > Delete cache" and do it.
2) Click "Configure"

The configuration dialog provides options for Generator and Platform, such as the compiler to use or to build 32 bit instead of the default 64 bit.

Credit

Thanks and credit for the CMake files which were first developed and contributed by Alexandre Buge (https://github.com/Qlex42) and revisions by Jean-Michaël Celerier (https://github.com/jcelerier) and Matti Regenhardt (https://github.com/MattiRegenhardt).