Building SheafSystemProgrammersGuide™ on Windows

1 Copyright Notice

2 Platform

SheafSystemProgrammersGuide, the tutorial suite for the SheafSystem™ libraries, is supported for Windows 10 using Visual Studio 2015.

3 Software Prerequisites

- CMake 3.5.0. Cross platform build tool; down load from www.cmake.org.
- Visual Studio 2015. C++ compiler; available from Microsoft.
- 7-zip 9.20 or newer. File archiving utility; down load from www.7-zip.org.
- SheafSystem, installed libraries or source tree.

Binary distributions with Windows installers are available for CMake and 7-zip; just use the default installation procedure.

4 Building and running SheafSystemProgrammersGuide

Step 1: Extract the package in a directory of your choice.

We'll assume you've down loaded the source as a zip file, SheafSystemProgrammersGuide-<version>.zip. (For instance, the SheafSystemProgrammersGuide Github page provides a link to download SheafSystem-<branch or tag>.zip). In Windows Explorer, with 7-zip installed, just navigate to the folder containing the down load and right-click on the file and select one of the extraction options to extract into a location of your choice. The package will extract into <your choice>\SheafSystemProgrammersGuide-<version>. From here on we'll refer to that location as <programmers_guide_source>

Step 2: Configure with CMake

Start the CMake application. In the "where is the source code" box enter or browse to <programmers_guide_source>/examples. (In CMake all paths are displayed with forward slashes.) In the "where to build the binaries" box enter <programmers_guide_source>/examples/build. Click on the configure button, then click the yes button to create the build directory and click the finish button to accept the default generator (Visual Studio). Configuring will start, it may take a while. Typically, a message will be displayed about not being able to find the prerequisites.

There are two check boxes in the upper right part of the display. The "Advanced" check box toggles the display between "basic" and "advanced" mode. In basic mode the display shows only the variables you need to set to configure the system. In advanced mode, the display shows a large number of variables detailing the configuration process. Toggle the display to basic mode.

The "Grouped" check box toggles the display between "ungrouped" and "grouped" modes. In "ungrouped" mode the variables are listed in alphabetical order; in "grouped" mode, the variables are organized into an outline-like display. Choose whichever mode you find most appealing, but if in "grouped" mode, expand the headings so that all variables are visible.

There is only one group of variables you need to review, and perhaps set, to configure the SheafSystemProgrammersGuide: the PREREQ_ variables.

Step 3: Set the PREREQ_ variables.

The PREREQ_ variables control the search for the prerequisites. There are three methods for setting these variables: direct entry, command line entry, and environment variable entry. Direct entry and command line entry are as described above. To use environment variable entry, set an environment variable of the same name to the desired value before invoking ccmake. Note that no matter which of these methods is used, it is important to set the value correctly. Incorrect values may produce unpredictable and hard to interpret results. In this case, it is often best to just delete the build directory and try again from Step 5!

There is one PREREQ variable:

PREREQ_SHEAFSYSTEM_CONFIG_DIR (type PATH): the absolute path to the directory containing the file SheafSystemConfig.cmake. For instance, <installation tree root>/cmake or <source tree root>/build/cmake.

Click the configure button again. If it completes without error, click the generate button. Otherwise, correct the variable as needed and click configure again. When you've successfully generated, you're done, exit CMake.

Step 4: Build the example executables

In the Windows Explorer navigate to <programmers_guide_source>\build. Double click on SheafSystemProgrammersGuide.sln to start Visual Studio. In Visual Studio select Debug_contracts in the Solution Configurations box. (Debug_contracts is the only supported

configuration.) In the Solution Explorer pane, right click on the "ALL_BUILD" target and select Build. Visual Studio will build all the example executables.

Step 5: Run the examples

Some of the examples depend on the output of preceding examples, so you have to execute the examples in order. To run an example, right click an the associated target, select "Set as Startup Project", then type <Cntl-F5>. The example will run in a console window, then prompt you to "press any key to continue …". When you press any key, the console window will close.