Knossos Build -Step by Step Guide

1) Downloaded Knossos Git release tag aeb3f79(Migrated to Knossos Release 5).

2) Installed msys from <https://sourceforge.net/p/msys2/wiki/MSYS2%20installation/>.

We use the MSYS(MINGW) cross compilation tool to build.

3) Start the MSYS2 Shell from the programs menu. (There is a different shell to run and

build called MinGW-w64 Win64 Shell and a separate one to install packages to the

MinGW build environment called MSYS2 Shell.)

4) In the MSYS2 shell install the dependencies with pacman with following command:

 pacman -S.

5) Install the packages

* **qt5:** mingw-w64-x86\_64-qt5
* **boost:** mingw-w64-x86\_64-boost
* **freeglut:** mingw-w64-x86\_64-freeglut
* **libcurl:** mingw-w64-x86\_64-curl
* **google snappy:** mingw-w64-x86\_64-snappy.
* **quazip:** mingw-w64-x86\_64-quazip
* **Quazip and PythonQt:** download and install the x86\_64 versions of the unofficial packages from the CDCU share with pacman -U [package name].

If the package names have changed, you can always look for the correct package name with

pacman - Ss [substring], e.g. pacman -Ss freeglut

6) The project uses CMake to generate the makefiles. The IDE used for the build is QTcreator.

Using msys installation shell install the required tools for the build depending on the platform.

For eg: Install cmake, g++, qtcreator, python for mingw-w64-x86\_64 platform as this is our

underlying platform. Open the qtcreator from the msys installation folder under

- mingw64/bin/qtcreator.exe using the MINGW-w64 WIN64 Shell.

7) **Before we run cmake, setup a toolkit to configure the build the environment. Under the Tools/Build &**

**Run/Kits/Add/ create a configuration kit of your choice name and choose the tools as follows:**

* Compiler – msys2 MINGW - x86\_64 msys64/mingw64/bin/g++.exe
* Qt version - Qt MINGW-w64 64bit (MSYS2) – msys64/mingw64/bin/qmake.exe
* Cmake – mingw x86\_64 cmake – msys64/mingw64/bin/cmake.exe

**8) Remove c:\Windows\System32\python27.dll in**

**C:\msys64\mingw64\lib\cmake\Qt5Python27\Qt5Python27Targets-release.cmake**

**C:\msys64\mingw64\lib\cmake\Qt5Python27\_QtAll\Qt5Python27\_QtAllTargets-release.cmake**

Add the following under target\_link\_libraries:

${PYTHON\_LIBRARIES}

Open the Knossos CmakeLists.txt as a project file in Qtcreator. Choose the previously created kit

for setting up the build configuration. Run cmake without passing any additional arguments.

After the project has configured then build. (This should build the Knossos release 4.1.2. Please note no additions have been made to the source code.)

9) The vtk package in mingw has hard coded paths for linking against its libraries pointing to the

directory “C:/building/msys64/mingw64” which does not actually exist. Need to run the following

commands to get rid of this bug.

cd /mingw64/lib/cmake/

find . -name '\*.cmake' -exec grep -H 'C:/building/msys64' {} \;

find . -name '\*.cmake' -exec sed -s "s|C:/building/msys64|C:/msys64|g" -i {} \;

find . -name '\*.cmake' -exec grep -H 'C:/building/msys64' {} \;

Do this step if you install mingw vtk (optional) . If you choose to build vtk from source do not install mingw vtk package.

**10) Download VTK-7.0.0 git source and sync to tag**

**Set the following arguments before running cmake.**

-DVTK\_Group\_Qt:BOOL=ON -DBUILD\_SHARED\_LIBS:BOOL=ON -DCMAKE\_BUILD\_TYPE=Release -DCMAKE\_INSTALL\_PREFIX=C:/msys64/mingw64.

**(If the cmake build fails complaining about g++ compiler version 6, change line 47 in**

**vtkCompilerExtras.cmake from**

string (REGEX MATCH "[345][\\.[0-9]\\.[0-9]\*](file:///\\.[0-9]\\.%5b0-9%5d*)" to

string (REGEX MATCH "[3456]\\.[0-9]\\.[0-9]\*"(optional))

**After the build is complete, go to build tab and then click on the “Deploy project VTK” option.**

**11) Install hdf5 package from mingw.**

12) Make a few additions to the Knossos CmakeLists.txt. Comment line 31 in the text file.

Add the following under target\_link\_libraries:

${VTK\_LIBRARIES}

${SZIP\_LIBRARIES}

${HDF5\_LIBRARIES}

Open the Knossos CmakeLists.txt as a project file in Qtcreator. Choose the previously created kit

for setting up the build configuration. Run cmake without passing any additional arguments.

After the project has configured then build. This version of Knossos release 4.1.2 has additional vtk and hdf5 capabilities added to the source code.

13) In order to deploy the project, use the ldd\_cp.sh present in share under CDCU\common\for\_rutuja.

Copy the script to the build folder and run passing it the following arguments.

./ldd\_cp.sh -executable name -deploy folder

Inside the deploy folder all the program specific libraries will get copied. Place the executable in this

folder. Create a subfolder within this folder called *platforms*. Find the library qwindows.dll from the

mingw environment setup and copy it into the *platforms* folder. Zip this folder to any machine and run

the executable on that machine.

(The above dynamic build setup is applicable only for the Knossos release tag aeb3f79 – release 4.1.2. For other releases or git tags of Knossos, we might be able to get a static build.)