Installing Qt

Author: Zia Khan
Email: zia@cs.umd.edu

NOTE: This document has several trouble-shooting tips. Please see if this document contains the problem/error you encountered before emailing the instructor.

For Windows

1. Check for any Windows updates using the Windows Update Center and check for any graphics drivers updates if you have a dedicated graphics card
2. Download Microsoft Visual Studio 2013 (Community Edition) for free here: https://www.visualstudio.com/en-us/news/releasenotes/vs2013-community-vs
3. Go to http://www.qt.io/download-open-source/ click show downloads and select an offline installer for the latest stable release of Qt (v5.7 at this time) for Windows 64-bit (VS 2013, OpenGL, XXX MB)
4. Run the installer.

For Linux

1. Check for any system updates
2. Install and/or update gcc using the Ubuntu equivalent of sudo apt-get install gcc and sudo apt-get update gcc
3. Go to http://www.qt.io/download-open-source/ and click the online installer for Linux
4. Run the installer

For OSX

1. Install Xcode.
2. Run Xcode and accept the licensing agreement if you haven't already.
3. Run "xcode-select --install" to install the command line tools.
4. Go to https://www.qt.io/download-open-source, download and run the installer for MacOSX.

Using the Command Line

For Windows

1. Look for where the Qt installer put qmake.exe. It will be in a directory similar to "C:\Qt\Qt5.x\5.x\msvc2013_64_opengl\bin". Take note of this path.
2. Go to Control Panel | System and Security | System. Click on Advanced Systems settings. Click Environment Variables and find PATH under System variables. Double-click it and add the Qt path from the previous step (";C:\Qt\Qt5.x\5.x\msvc2013_64_opengl\bin" replacing the x's with your version of Qt) to the end of your PATH variable.
3. Open the Visual Studio Native Tools Command Prompt. If you do not have a shortcut to it, hit the windows key (in between the fn and alt keys on most keyboards) and type "command prompt" and click on the one that resembles "Open VS2013 x64 Native Tools Command Prompt"
4. You can now navigate to the project folders and run qmake and nmake (Visual Studio's version of make). Reminder: dir is the equivalent of ls in windows.

For Linux/OSX

1. Look for where the Qt installer put the qmake. Add this directory to your path (e.g /Users/zia/Qt /5.x/clang_64 for MacOSX or /home/zia/Qt /5.x/gcc_64/bin for Linux). The x.x's are just subversion numbers of Qt. You want to be using 5.3 or newer.
2. To change your PATH variable, edit the .profile or .bash_profile file in your home directory. Add this line at the end of the file to add another directory to PATH: export PATH="$PATH:*your_directory_containing_qmake*"
3. To compile your application, in the directory containing the .pro file. Run qmake and then make. NOTE: If you have trouble running qmake or building using Qt under Linux. Try the following https://qt-project.org/wiki/Install_Qt_5_on_Ubuntu, installing additional OpenGL libraries as well as the g++ compiler.

Using Qt-Creator

When you first open Qt-Creator, it will prompt you to configure a compiler and the Qt version. It should automatically detect the VS compiler in Windows and GCC in Linux along with the correct version of Qt.

A few notes on Qt-Creator

- It resembles Eclipse, but is much simpler than it
- The execute, debug, and build icons are located in the bottom left hand corner
- By default, the compiled files and executable are placed in the debug folder. You can change it to the release folder by clicking on the computer icon located above the execute button.
- You can create a new project under File | New File or Project. The easiest option is to choose Other Project | Empty Qt Project to avoid any unnecessary files or inclusions.
- To add source and header files, right click on the project in the left pane and choose Add New. Here, you can easily add header and source files simultaneously by choosing C++ class as the type of file. For shaders, choose GLSL | Vertex Shader (or Fragment Shader). For shaders, make sure the file extension is .vsh for vertex shaders or .fsh for fragment shaders so that the IDE highlights variables, functions, etc. correctly for those files.

Running Applications with Command-line Options

For those of you that are having some trouble with the command-line (in particular, the Windows users), there is an alternative way you can run your code from within Qt Creator with command-line arguments. Here are the instructions:

1) With the project open in Qt Creator, click on the "Projects" tab on the left-hand side

2) Near the top of the window that comes up, there will be two buttons that say "Build" and "Run". Click on the "Run" button.

3) You will see a text field called "Arguments". This is where you can type in the arguments to be passed into the program when you run it. For example, I could put

cmsc427.hdr output.hdr crop 100 100 1000 500

into the text field and then click the run button.

Addressing Could not resolve SDK path for 'macosxXX.X'

If you recently upgraded to the latest version of Mac OS, then the following message may apply to you. If you are trying to compile your code and are getting the error message

Could not resolve SDK path for 'macosx10.x'

Then try this fix.
Edit the file at <Path_to_Qt_Install>/5.x/clang_64/mkspecs/qdevice.pri
Comment out the following line by putting a hashtag (also called pound) in front of it like so

#!host_build:QMAKE_MAC_SDK = macosx10.x

Setting up Break Points and Line-by-Line Debugging

Change the release into debug in the second line of the the .pro file:

CONFIG += qt warn_on debug embed_manifest_exe

And also delete the two lines:

```
win32 {
  QMAKE_CFLAGS += "/D_HAS_ITERATOR_DEBUGGING=0 /D_SECURE_SCL=0"
  QMAKE_CXXFLAGS += "/D_HAS_ITERATOR_DEBUGGING=0 /D_SECURE_SCL=0"
}
```

Finally you should remove the Shadow Build option in the project configuration.
Then you could easily insert breakpoints and debug your program!!

Issues with PATH variables

If you have the problem of not being able to run nmake, double check your PATH variable. When you add the Qt directory to your PATH, do not delete the stuff that is already there. For example your PATH variable should not look like this

PATH=C:/QT/5.5/msvc2013/bin

We only want you to add the Qt directory to the already existing PATH variable, so it should look something similar to

PATH=%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wbem;C:/QT/5.5/msvc2013/bin

Debug/Release Mix Up Errors

If you get the following error from windows:

Return 0x527 from link.exe

states ITERATOR_DEBUG_LEVEL value 2 doesn't match value 0 in GLview.obj during the nmake process.

Try running

nmake distclean
qmake
nmake

This error occurs because .OBJ files from a release version are getting mixed with a debug version.

Unknown debugger type "No engine" Error Occurs When Trying to Debug

Visual Studio 2013 does not come with debugger that Qt can use installed by default. You might need to download and install the Windows SDK. I believe you need to link the CBD (command debugger), as WinDbg does not work.

Here is a link
https://msdn.microsoft.com/en-us/library/windows/hardware/ff551063%28v=vs.85%29.aspx

Alternatives to Mac Lab on Campus (McKeldin Library)

For those students that don't have an up-to-date graphics card and have to use a different computer to work on your projects, I have found a viable solution if you cannot use one of the Mac labs. You may use the computers in McKeldin library. Visual Studio 2013 is already installed. You just have to install Qt to begin working on your projects. I've written up a small set of instructions to follow if you use the library computers.

1 Download/Install the Qt offline installer (Windows 64 bit for Visual Studio) from here.
2 Add qmake as an environment variable. To do this:
- Go to the system properties window where environment variables are listed. Under "User variables for <your UMD directory ID>", add a new environment variable. To do this, click "new" and enter the following information
- Variable Name: PATH
- Variable Value: C:\Qt\Qt5.5.0\5.5\msvc2013_64\bin
- Click OK and close the window.

3 Visual Studio 2013 is already installed (which means nmake is installed). The console you need to use is located here
- C:\Program Files (x86)\Microsoft Visual Studio 12.0\Common7\Tools\Shortcuts\VS2013 x64 Native Tools Command Prompt

4 Before running qmake and nmake, remove the following line from the .pro file in the starter code
- QMAKE_CXXFLAGS_WARN_ON += -Wno-unknown-pragmas

As long as you use the offline Qt installer, you should be able to install Qt without any special privileges needed by the IT help desk. While I did not test every computer in the library, I have

checked and was able to run everything on the Windows computers on the 1st floor.

Note - These are public computers. They're rebooted on a daily basis so you'll lose any work if it's saved locally. This means you'll need to reinstall Qt each time you use the computer. I advise using a flash drive and keeping a copy of the Qt offline installer on it as well as your project code.

xcodebuild: error: The directory does not contain an Xcode project.

From: http://stackoverflow.com/questions/39492617/xcode-8-error-project-error-xcode-not-set-up-properly-even-though-the-licens
and
http://stackoverflow.com/questions/33728905/qt-creator-project-error-xcode-not-set-up-properly-you-may-need-to-confirm-t

Find the folder where you install Qt.
Open in a text editor the file at
Qt_install_folder/5.7/clang_64/mkspecs/features/mac/default_pre.prf
Find the line with text (for me it was line 15):

isEmpty($$list($$system("/usr/bin/xcrun -find xcrun 2>/dev/null")))): \
Replace line with:
isEmpty($$list($$system("/usr/bin/xcrun -find xcodebuild 2>/dev/null"))): \
Save & re-compile