Installation Cheat Sheet - OpenCV 2.4.11 Compile From Source and Qt using Windows 7 + Qt ???

(should also work with Windows 8/8.1, not tested though)

download and install Visual Studio 2013 Community Edition (yes, its free) (choosing all default options will work fine)

download the latest 32-bit version of Qt, for example:
qt-opensource-windows-x86-msvc2013_opengl-5.4.1.exe
(Note: do NOT use the Qt Online Installer, this will download the 64-bit version if you have a 64-bit computer)
(Note 2: on the Qt page when you get to "Download Now", scroll down further to "View All Downloads",
this should list the Offline Installer for the 32-bit Windows Visual Studio compiler version of Qt as a choice)

download OpenCV 2.4.11

make a folder "C:\OpenCV-2.4.11" and extract OpenCV 2.4.11 to there
make a folder "C:\OpenCV-2.4.11\mybuild"

download the latest version of CMake for windows with the Windows 32-bit Installer,
for example "cmake-3.2.2-win32-x86.exe"

during the CMake installation, choose the option "Add CMake to the system PATH for all users"

reboot, start CMake

set "Where is the source code:" to C:/OpenCV-2.4.11/opencv/sources
set "Where to build the binaries:" to C:/OpenCV-2.4.11/mybuild

press Configure, choose "Visual Studio 12 2013" from the drop-down menu
(do NOT choose the 64-bit option, which is titled "Visual Studio 12 2013 Win64")
choose the "Use default native compilers" radio button, then choose Finish
after a moment you will get a list of options, all in red
scroll towards the bottom and check "WITH_QT", the press Configure again
after another moment the previous lines should now be white, with new lines pertaining to Qt only in red
press Configure a 3rd time, after a moment all lines should now be white
press Generate

when generating is done, in "C:\OpenCV-2.4.11\mybuild" there should be a file "OpenCV.sln"
this is a regular Visual Studio 2013 solution file, double click to open in Visual Studio
verify "Solution Configurations" and "Solution Platforms" are set to "Debug" and "Win32" respectively,
then choose:
Build -> Build Solution

you will likely get multiple warnings and a linker error pertaining to "python27_d.lib",
as long as there are no other errors its ok

remove any OpenCV directories in your PATH currently, for example if you added
"C:\OpenCV-2.4.11\opencv\build\x86\vc12\bin" when following part 1 then remove that at this time

add the updated **bin** directory to the operating system PATH:
C:\OpenCV-2.4.11\mybuild\bin\Debug
pull up Command Prompt and verify the updated **bin** directory (and no other OpenCV directories) is now in PATH, then reboot
(note that the **bin** directory is *different* from the precompiled binary directory used in part 1)

start Visual Studio 2013, make a new project
choose Visual C++, Win32 Console Application, name as you prefer, ex "SimpleCanny1"
set preferred location, uncheck "Create directory for solution" and "Add to source control", choose OK,
choose Next, uncheck "Precompiled Header" and "Security Development",
check "Empty Project" and verify "Console application" radio button is checked
choose Finish

in VS go to:
Project -> Properties -> Configuration Properties -> VC++ Directories -> Include Directories
add the include directory: C:\OpenCV-2.4.11\opencv\build\include
(note that the **include** directory is *the same as* the precompiled binary directory used in part 1)

in VS go to:
Project -> Properties -> Configuration Properties -> VC++ Directories -> Library Directories:
add the **library** directory: C:\OpenCV-2.4.11\mybuild\lib\Debug
(note that the **library** directory is *different* from the precompiled binary directory used in part 1)

in Windows Explorer (not within Visual Studio) navigate to the **lib** directory:
C:\OpenCV-2.4.11\mybuild\lib\Debug
verify the libs listed are the same as this list:

opencv_calib3d2411d.lib
opencv_contrib2411d.lib
opencv_core2411d.lib
opencv_features2d2411d.lib
opencv_flann2411d.lib
opencv_gpu2411d.lib
opencv_haartraining_engined.lib
opencv_highgui2411d.lib
opencv_imgproc2411d.lib
opencv_legacy2411d.lib
opencv_ml2411d.lib
opencv_nonfree2411d.lib
opencv_objdetect2411d.lib
opencv_ocl2411d.lib
opencv_photo2411d.lib
opencv_stitching2411d.lib
opencv_superres2411d.lib
opencv_ts2411d.lib
opencv_video2411d.lib
opencv_videostab2411d.lib

then in VS copy/paste this list of libs into:
Project -> Properties -> Configuration Properties -> Linker -> Input -> Additional Dependencies

next in the Visual Studio toolbar, verify that "Solution Configurations" and "Solution Platforms" are set to "Debug" and "Win32", respectively

next, copy any JPEG image into the project directory and rename it "image.jpg"
(unless you are going to use a webcam feed, in which case this is not necessary, see below)

from my MicrocontrollersAndMore GitHub page, copy/paste CannyStill.cpp (uses a still image), CannyWebcam.cpp (uses a webcam), or RedBallTracker.cpp (tracks a red ball, uses a webcam) and run (with or without debugging)

now we will move on to Qt: