# How to setup Qt and openCV on Windows

From Qt Wiki
Jump to: navigation, search

How to setup Qt and openCV on Windows

## Introduction



This article shows how to install Qt, build OpenCV, and run a basic OpenCV example. This article assumes Windows 10 has just been installed.

This procedure requires close to 10GB of disk space:

```
Qt: 5.06GB
opencv: 522MB
opencv-Build: 3.95GB
downloads:152MB
```

This article uses information from the following pages: http://docs.opencv.org/2.4/doc/tutorials/introduction/windows_install/windows_install.html?highlight=installation http://www.laganiere.name/opencvCookbook/chap1s1_2.shtml

## Windows 10, Qt 5.9, OpenCV 3.2.0

This guide is actual for Qt 5.12.2 with MinGW 7.3.0 and OpenCV 4.0.1 too

### Qt

#### Installation

Download the Qt installer from www.qt.io (https://www.qt.io/download-open-source/), then choose "Download now". This will then download qt-unified-windows-x86-2.0.5-online.exe. Execute the program, then choose the following settings:

```
Welcome to the Qt online installer: next
Qt Account - your unified login to everything Qt: skip
Setup-Qt: next
installation folder: D:\Qt
select components: Qt-Qt5.9-MingGW 5.3.0 32 bit
select components: Qt-Tools-MinGW 5.3.0
License Agreement: agree and next
start menu shortcuts: next
ready to install: install
```

#### Testing

Run D:\Qt\Tools\QtCreator\bin\qtcreator.exe

```
File-New file or project-Qt Widgets Application-choose
enter a name and a location: next
select all kits: next
Class information: MainWindow (defaults): Next
Project management: Finish
```

Now a new project is made. Start debugging by choosing

```
Debug-Start Debugging-Start debugging (F5)
```

Now the Qt tab in the Windows taskbar should turn into a progress bar. After some time a new empty window should pop up. Stop debugging either by pressing the red cross in the top right of this new window, or choose

```
Debug-Stop debugging
```

#### Adjust Qt

When you need to add, remove or update a component of Qt, this can be done by running D:\Qt\MaintenanceTool.exe:

```
maintain Qt: Qt Account: Skip
Setup Qt: Add or remove components: Next
```

```
Select components:
next :update
```

## Cmake

Download cmake from cmake.org (https://cmake.org/download/). In this guide, 3.7.2 (https://cmake.org/files/v3.7/cmake-3.7.2-win64-x64.msi) is used. Start cmake-3.7.2-win64-x64.msi, then choose the following settinge:

```
Welcome to the CMake etup Wizzard: next
End-User License Agreement: [X] Accept and next
Install options: [X] Add CMake to the system PATH for all users, next
Destination folder: C:\Program Files\CMake (default), next
Ready to install CMake, Install
```

## OpenCV

### Getting OpenCV

Download openCV from sourceforge (https://sourceforge.net/projects/opencvlibrary/). In this guide, version 3.2.0 (https://sourceforge.net/projects/opencvlibrary/files/opencv-win/3.2.0/) is used. Start opencv-3.2.0-vc14.exe and let it extract to d:\ . Now the folder d:\opencv is created.

### Add minGW to the windows PATH variable

```
Open the control panel,
System and Security,
System,
Advanced system settings,
Environment Variables,
System Variables,
Variable Name: Path
Variable value: ;D:\Qt\Tools\mingw530_32\bin
```

### Compiling OpenCV

Start C:\Program Files\CMake\bin\cmake-gui.exe then choose the following settings:

```
Where is the source code: D:\opencv\sources
where to build the binaries: D:\opencv-build
```

Then click Configure, let cmake create the build directory, and choose the following settings:

```
Specify the generator for this project: MinGW Makefiles
Specify native compilers, next
Compilers C: D:\Qt\Tools\mingw530_32\bin\gcc.exe
Compilers C++: D:\Qt\Tools\mingw530_32\bin\g++.exe
Finish
```

```
Check the box [X]WITH_QT
Check the box [X]WITH_OPENGL
set Qt5_DIR to D:\Qt\5.9\mingw53_32\lib\cmake\Qt5
```

```
Uncheck the box []ENABLE_PRECOMPILED_HEADERS
```

Then click configure again.

```
Set QT_MAKE_EXECUTABLE to D:\Qt\5.9\mingw53_32\bin\qmake.exe
Set Qt5Concurrent_DIR to D:\Qt\5.9\mingw53_32\lib\cmake\Qt5Concurrent
Set Qt5Core_DIR to D:\Qt\5.9\mingw53_32\lib\cmake\Qt5Core
Set Qt5Gui_DIR to D:\Qt\5.9\mingw53_32\lib\cmake\Qt5Gui
Set Qt5Test_DIR to D:\Qt\5.9\mingw53_32\lib\cmake\Qt5Test
Set Qt5Widgets_DIR to D:\Qt\5.9\mingw53_32\lib\cmake\Qt5Widgets
Set Qt5OpenGL_DIR to D:\Qt\5.9\mingw53_32\lib\cmake\Qt5OpenGL
Set CMAKE_BUILD_TYPE to Release or RelWithDebInfo
Set OPENCV_VS_VERSIONINFO_SKIP=1
```

Then click configure again Then click generate

Next open cmd, and type the following commands. To speed up the compile, the -j flag can be used to run multiple compile jobs simultaneously. On an 8 core CPU, you can set it to 8 or higher, so all cores are used. On a core i7-3770@3.4GHz with 8GB ram, the compile takes about 6 minutes.

```
d:
cd d:\
cd opencv-build
mingw32-make -j 8
mingw32-make install
```

If, in the file opencv/sources/modules/videoio/src/cap_dshow.cpp, you have the following error : 'sprintf_instead_use_StringCbPrintfA_or_StringCchPrintfA' was not declared in this scope ...

try this :put the following line: #define NO_DSHOW_STRSAFE, before the line : #include "DShow.h"

If you have the error: ‘nullptr’ was not declared in this scope..

try this: in cmake check the box ENABLE_CXX11

If, in the file modules\videoio\src\cap_msmf.cpp you have the error: using invalid field '{anonymous}::ComPtr<T>::p'..

try this: in cmake unchecking WITH_MSMF

### Add OpenCV compiled libraries to the windows PATH variable

```
Open the control panel,
System and Security,
System,
Advanced system settings,
Environment Variables,
System Variables,
Variable Name: Path
Variable value: ;D:\opencv-build\install\x86\mingw\bin
```

## Compile and run the example

Run D:\Qt\Tools\QtCreator\bin\qtcreator.exe

```
File-New file or project-Qt Widgets Application-choose
enter a name and a location: next
select all kits: next
Class information: MainWindow (defaults): Next
Project management: Finish
```

Now a new project is made.

modify the .pro file like this:

```
#-------------------------------------------------
#
# Project created by QtCreator 2017-03-05T12:30:06
#
#-------------------------------------------------

QT       += core gui

greaterThan(QT_MAJOR_VERSION, 4): QT += widgets

TARGET = opencvtest
TEMPLATE = app

# The following define makes your compiler emit warnings if you use
# any feature of Qt which as been marked as deprecated (the exact warnings
# depend on your compiler). Please consult the documentation of the
# deprecated API in order to know how to port your code away from it.
DEFINES += QT_DEPRECATED_WARNINGS

# You can also make your code fail to compile if you use deprecated APIs.
# In order to do so, uncomment the following line.
# You can also select to disable deprecated APIs only up to a certain version of Qt.
#DEFINES += QT_DISABLE_DEPRECATED_BEFORE=0x060000    # disables all the APIs deprecated before Qt 6.0.0


SOURCES += main.cpp\
        mainwindow.cpp

HEADERS  += mainwindow.h

FORMS    += mainwindow.ui

INCLUDEPATH += D:\opencv\build\include

LIBS += D:\opencv-build\bin\libopencv_core320.dll
LIBS += D:\opencv-build\bin\libopencv_highgui320.dll
LIBS += D:\opencv-build\bin\libopencv_imgcodecs320.dll
LIBS += D:\opencv-build\bin\libopencv_imgproc320.dll
LIBS += D:\opencv-build\bin\libopencv_features2d320.dll
LIBS += D:\opencv-build\bin\libopencv_calib3d320.dll

# more correct variant, how set includepath and libs for mingw
# add system variable: OPENCV_SDK_DIR=D:/opencv/opencv-build/install
# read http://doc.qt.io/qt-5/qmake-variable-reference.html#libs

#INCLUDEPATH += $$(OPENCV_SDK_DIR)/include

#LIBS += -L$$(OPENCV_SDK_DIR)/x86/mingw/lib \
#        -lopencv_core320        \
#        -lopencv_highgui320     \
```

```
#        -lopencv_imgcodecs320   \
#        -lopencv_imgproc320     \
#        -lopencv_features2d320  \
#        -lopencv_calib3d320
```

and modify mainwindow.cpp like this:

```
#include "mainwindow.h"
#include "ui_mainwindow.h"

#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);

    // read an image
    cv::Mat image = cv::imread("f://1.jpg", 1);
    // create image window named "My Image"
    cv::namedWindow("My Image");
    // show the image on window
    cv::imshow("My Image", image);
}

MainWindow::~MainWindow()
{
    delete ui;
}
```

Place an image with the name "1.img" in the root of F:\, then run the example. Now 2 windows should pop up. One with the image, and one with an empty window.

Retrieved from "https://wiki.qt.io/index.php?title=How_to_setup_Qt_and_openCV_on_Windows&oldid=36678"

---

- This page was last edited on 24 January 2020, at 20:20.