Archived:How to install and use the Qt SDK Remote Compiler

Archived: This article is **archived** because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

The Qt Remote Compiler service is no longer available.

This article explains how to set up and use the Archived:Qt SDK Remote Compiler on different platforms.

Introduction

18 Jul 2010

The Archived:Qt SDK Remote Compiler is very useful if you want to develop for Symbian on a platform that doesn't have the necessary toolchain (e.g. Linux or Apple). It uses a compilation service provided by Nokia Developer, creating a simple, standardized environment for building Qt applications and creating installation packages for Symbian and Maemo. The Nokia Remote Compiler is supported in release 1.0 of the Qt SDK. This tutorial article requires the install of the latest release version initial release date 23-Jun-2010.

Ubuntu 10.04 Remote Compiler installation

Those people wanting to use the Remote Compiler in using **Ubuntu 10.04 LTS** operating system built on Linux kernel should load the **offline** version of the SDK. When the download is complete right click on the file "open containing folder" from the Firefox Download window ,right click on the file *Nokia_Qt_SDK_Lin32_offline_v1_0_en* select "Properties" and Select the *Permissions* tab tick the box "Execute". Click "Close" and double click the file to start SDK installation.

The Ubuntu version of Nokia Qt SDK will start select **Customise** in the installation options and select **Experimental API's**. This will select the Remote Compiler installation. The online version does *not* currently support installation of the Remote Complier and is a known problem. If you want to check the install log whether the Remote Compiler is installed, this is the last thing to be installed and displays as "Installing Remote Compiler Plugin" at around 98% of the Ubuntu version of SDK install. If you already installed the online version you need to reinstall Ubuntu version of Nokia SDK just delete the directory in your home directory to uninstall before installation of a new copy, because the SDK has not been installed into the Ubuntu installer. Follow the instructions below ignoring the External API configuration.

Ubuntu 10.04 Nokia Qt SDK and Remote compiler installation

From Nokia Qt SDK 1.0.1 onward, the install kit will give an error message "C++ compiler not installed". The C++ compiler is not installed by default on an Ubuntu 10.04 installation. Open a Terminal window, by Applications -> Accessories ->Terminal, and type the following command to the bash shell "**sudo apt-get install g++**". The g++ complier will automatically install. This will now allow the Nokia Qt SDK to install. Click "Custom" and make sure the install path in the window is /home/yourUSERNAME. In the next box tick the Remote Compiler box and then Next to continue the installation.The use of the Simulator allows to test the examples before sending to the Remote Compiler and can identify and potential problems. There is a restriction of 1200 seconds, 20 minutes, to complete compilation of project remotely, including to time to upload the source code.

Update 21-Oct-2010 :- *Nokia Qt SDK 1.0.1 works on Ubuntu 10.04 but there are failures,bus errors, running under Ubuntu 10.10 beta. The suggestion is to install Ubuntu 10.04 archived and stable release and upgrade to 10.10 later if needed. Ubuntu 10.04 has support until 2012 and is obtainable on the drop down release selector on the Ubuntu download page.*

Creating an Ubuntu desktop launcher (shortcut) for Qt Creator

There isn't any desktop launcher (shortcut like in Windows) during installation so you will need to create your own. Right click on the Ubuntu desktop. Select "Create Launcher". In the "Name" box type "Qt Creator".
Click "Browse..." and browse to the path " /home/<USERNAME>/NokiaQtSDK/QtCreator/bin/" click on "qtcreator.bin", click "open", in the "Comment" box type Qt Creator. To set the icon to Qt, click on the default picture of spring navigate to path /home/<USERNAME>/NokiaQtSDK/readme/images/ scroll down to "qt_logo.png",click "open", back in the "Create Launcher" click "OK" and the launcher (shortcut) is created on the desktop ready for use below.

Configuring the Ubuntu 10.04 QEMU emulator for Maemo

This is for **Ubuntu 10.04 Linux 32** only. Although not part of the Remote Compiler this often causes problems for those with little experience of Linux system. The QEMU is installed in Nokia Qt SDK Qt Creator 2.0,but you cannot see the start icon until the "xserver-xephyr" is installed.

The way to do this is to install from Ubuntu launchpad website. The file to download is the xserver-xephyr i386.deb. When installed you need to start a terminal from the Applications -> Accessories -> Terminal and execute the following commands which are also described in the Wiki article Running Maemo 5 SDK (Linux) Further runs of the emulator will probably not require the commands

```
Xephyr :2 -host-cursor -screen 800x480x16 -dpi 96 -ac  &
export DISPLAY=:2
```

This will bring up a xserver window in black. When you start Qt Creator select you remote compiled project and go Tools-> Options -> Projects, Select the "Maemo Configurations" tab Add a new Configuration call this name "New Maemo 5 Emu" and click "Maemo emulator" instead of Remote Device, then click Password. Click "ok" and go back to main menu on the bottom of the left pane appears an icon, holding the mouse over this "Start Emulator". Click this and the emulator starts and you will see the picture in the bottom of Wiki article Running Maemo 5 SDK (Linux) showing the home screen like the one found in N900. This will initially take several minutes for the splash screen to display. Sometimes the QEMU screen needs a mouse click to display to picture. Click on the six blue boxes to get the N900 menu displayed. Once the emulator has fully started you go back to the "Maemo Configurations" tab and click "Test" to make sure your configuration is working. Hopefully you will get a listing similar to below.

```
Device configuration successful.
Hardware architecture: armv7l
Kernel version: 2.6.28-omap1
List of installed Qt packages:
...
list removed
...
```

In Qt Creator click the bottom button now showing "Stop Emulator" to finish the session.

Remote Complier Qt 4.6.2 sis files

Before you can run an application on a Nokia Mobile phone built with the Remote Compiler under Ubuntu you will need to download the Qt libraries to your phone and these can be downloaded from the Index of ftp://ftp.qt.nokia.com/pub/qt/symbian/4.6.2 website which must be installed on your Symbian based phone.The main file to load is fluidlauncher.sis which will request the download of the other library components. The fluidlauncher.sis must be able to run before trying to download your application. Remember to connect you phone to the wireless router or have an unlimited 3G data plan on your phone.

Troubleshooting Ubuntu

If you still get problems with missing libraries and various other problems in Terminal and Administrator give the command to install

```
sudo apt-get install libglib2.0-dev libSM-dev libxrender-dev libfontconfig1-dev libxext-dev
```

If you want to attempt to develop or use "QtOpenGL" you must have the libraries installed
This calls up several packages

```
sudo apt-get install libglu-dev
```


(C) Copyright Nokia 2014. All rights reserved.

```
The following NEW packages will be installed libdrm-dev libgl1-mesa-dev libglu1-mesa-dev
mesa-common-dev
```

See the link to Qt OpenGL on Qt website Qt OpenGL Examples

Later versions of Symbian Qt 4.7.1 binaries ftp://ftp.qt.nokia.com/pub/qt/symbian/4.7.1

Windows Remote Compiler

To follow this tutorial you will need to start Qt Creator 2.0 and Click **File** -> **Open File or Project...**
Select the following project path **C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector**
Select the file **simpleselector.pro**. The project will then load.

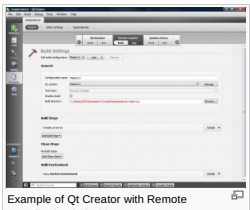
Installation

The Install and Configure of the Remote Compiler is given in the help file Building with the Remote Compiler.For convenience the important points are given below

Note: You can build applications from Qt Creator by setting up Remote Compiler as a build target.
This is an experimental component that you must install separately from the package that is included in the Nokia Qt SDK.
In the Nokia Qt SDK installation directory, double-click SDKMaintenanceTool.exe to install Experimental APIs.
In Qt Creator, choose Tools > Options > Projects > Remote Compiler.
Click the Terms of Service link to accept the terms.
In the User field, enter your Nokia Developer username and click Authenticate to log in to Nokia Developer.
Enter your password and click OK.

Configuration

- Choose **Projects** from the left pane
- Click the grey "+"
- From the "Add Target" drop down select **Remote Compiler**
- Click "OK" to add Remote Compiler as a target.
- In the new Remote Compiler box which has appeared at the top click **Build**


Example of Qt Creator with Remote compiler loaded with Maemo 5

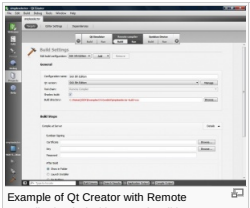
The **Configuration** box will display Maemo 5 and the **Qt version** box will display Maemo5

- Click the drop down box and select **S60 5th Edition**

The **Configuration** box does *not* update. Update this, if you do not replace "Maemo 5" name you will get "Duplicate Selection" or "Auswahl duplizieren" (Deutsch).
. This configuration name must be *unique*.

Therefore you must change the name in **Configuration** box Click on the "Maemo5" delete name an replace with **S60 5th Edition**

Now look at the **Edit build configuration** under "Build Settings" this has changed to **S60 5th Edition**


Example of Qt Creator with Remote compiler loaded with S60 5th Edition

- If you are using self signed certificate then ignore the next signing step. Currently there is no support to generate an unsigned sis file for the Symbian Open Signed Online website.
- If you have a Symbian developer signing certificate to add you must move down to build steps and click on the down arrow to the right of the words "Compile at Server". The details of certificate, key and password must be added.

Building the 'simpleselector' example

- Click to the Edit on the top left of Qt Creator 2.0, to take you back to you project.
- Click on the **4 Compile Output** along the bottom of screen.
- Click on **Build** from File menu at the top of screen.
- Click on **Build All'** from the sub-menu

Watch the "Compile Output" window at the bottom of screen, the messages will be something like:

```
Running build steps for project simpleselector...
Uploading to remote compiler
Uploaded 4 kB, waiting for build
```

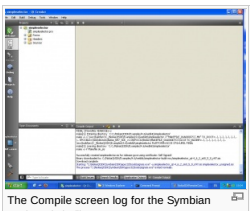
After a delay depending on the size of the project the output will appear.
The warning message below always appears for self-signed projects.

```
WARNING:
c:\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector\simpleselector.pro:15:
Unable to find file for inclusion \examples\symbianpkgrules.pri
bldmake bldfiles
ABLD.BAT build gcce urel
make -r -f
"\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector\EXPORT.make"
EXPORT VERBOSE=-s
make[1]: Entering directory `C:C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector'
Nothing to do
make[1]: Leaving directory `C:C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector'
make -r -f
"\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector\GCCE.make"
MAKEFILE VERBOSE=-s
make[1]: Entering directory `C:C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector'
make -s -C \wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector -f
"MAKEFILE_0XA000D7CC.MK" TO_ROOT=..\..\..\..\..\..\..\..\..
EPOCBLD=\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\MAKEFILE_0XA000D7CC\GCCE
TO_BLDINF=..\..\..\..\..\..\..\..\..\..\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector
PLATFORM=GCCE MAKMAKE
perl -S makmake.pl -D
\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector\SIMPLESELECTOR_0XA000D7CC
GCCE
make[1]: Leaving directory `C:C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector'
make -r -f
"\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector\GCCE.make"
LIBRARY VERBOSE=-s
make[1]: Entering directory `C:C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector'
make -s -C \wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector -f
"MAKEFILE_0XA000D7CC.MK" TO_ROOT=..\..\..\..\..\..\..\..\..
EPOCBLD=\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\MAKEFILE_0XA000D7CC\GCCE
TO_BLDINF=..\..\..\..\..\..\..\..\..\..\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector
PLATFORM=GCCE LIB
```

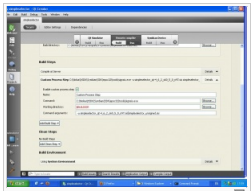

(C) Copyright Nokia 2014. All rights reserved.

```
make -s -r -f
"\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector\SIMPLESELECTOR_0XA000D7CC\GCCE\SIMPLESELECTOR_0XA000D7CC.GCCE"
LIBRARY
make[1]: Leaving directory `C:C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector'
make -r -f
"\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector\GCCE.make"
RESOURCE CFG=UREL VERBOSE=-s
make[1]: Entering directory `C:C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector'
make -s -C \wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector -f
"MAKEFILE_0XA000D7CC.MK" TO_ROOT=..\..\..\..\..\..\..\..\..
EPOCBLD=\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\MAKEFILE_0XA000D7CC\GCCE
TO_BLDINF=..\..\..\..\..\..\..\..\..\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector
PLATFORM=GCCE CFG=UREL RESOURCE
make -s -r -f
"\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector\SIMPLESELECTOR_0XA000D7CC\GCCE\SIMPLESELECTOR_0XA000D7CC.GCCE"
RESOURCEUREL
Creating
\s60\devices\s60_5th_edition_sdk_v1.0\epoc32\data\z\private\10003a3f\import\apps

Created \S60\devices\S60_5th_Edition_SDK_v1.0\epoc32\include\simpleselector.rsg

Creating
\s60\devices\s60_5th_edition_sdk_v1.0\epoc32\build\wcc\build\src\c_\nokiaqtsdk\examples\4.6\webkit\simpleselector\simpleselector_0xa000d7cc\gcce\urel
Creating \s60\devices\s60_5th_edition_sdk_v1.0\epoc32\release\gcce\urel
make[1]: Leaving directory `C:C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector'
make -r -f
"\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector\GCCE.make"
TARGET CFG=UREL VERBOSE=-s
make[1]: Entering directory `C:C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector'
make -s -C \wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector -f
"MAKEFILE_0XA000D7CC.MK" TO_ROOT=..\..\..\..\..\..\..\..\..
EPOCBLD=\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\MAKEFILE_0XA000D7CC\GCCE
TO_BLDINF=..\..\..\..\..\..\..\..\..\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector
PLATFORM=GCCE CFG=UREL BLD
make -s -r -f
"\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector\SIMPLESELECTOR_0XA000D7CC\GCCE\SIMPLESELECTOR_0XA000D7CC.GCCE"
UREL
main.cpp
window.cpp
moc_window.cpp
make[1]: Leaving directory `C:C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector'
make -r -f
"\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector\GCCE.make"
FINAL CFG=UREL VERBOSE=-s
make[1]: Entering directory `C:C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector'
make -s -C \wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector -f
"MAKEFILE_0XA000D7CC.MK" TO_ROOT=..\..\..\..\..\..\..\..\..
EPOCBLD=\S60\devices\S60_5th_Edition_SDK_v1.0\EPOC32\BUILD\MAKEFILE_0XA000D7CC\GCCE
TO_BLDINF=..\..\..\..\..\..\..\..\..\wcc\build\src\C_\NokiaQtSDK\Examples\4.6\webkit\simpleselector
PLATFORM=GCCE CFG=UREL FINAL
make[1]: Leaving directory `C:C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector'
make -s -f Makefile ok_sis

Successfully created simpleselector.sis for release-gcce using certificate: Self Signed!
Binary downloaded to: C:/NokiaQtSDK/Examples/4.6/webkit/simpleselector-build-
wcc/simpleselector_qt-4_6_2_s60_5_0.sis
Download complete
```

Generating a Symbian unsigned sis



The Compile screen log for the Symbian unsigned sis file



Adding Custom Process step for Symbian unsigned sis file

This only applies to Windows systems with the **Symbian SDK** loaded. There is a requirement to have an unsigned version of the sis file for use by Symbian Open Signed website with one IMEI. This is often carried out using the command **signsis -u symbian_selfsigned.sis symbian_unsigned.sis. This removes the self signed section of code**.

This all works well,with local SDK as the make file can have the commands added. When you need to Remote Compile you cannot change the Remote Compiler makefile. In Qt Creator 2.0 steps can be added after the remote compile is complete. The set up is best seen where the function "Add Build Step" and adding "Custom Process Step" in the project setup are used.

In here the full path to signsis.exe is added and in the "Command arguments" is the name of the compile remote signed sis file and the name of the unsigned sis file. There does not appear to be anyway of finding of the sis file name will be other than to test compile and see the name of the file returned. The name of the unsigned file is your choice.

Related documentation

- Building with the Remote Compiler
- Qt SDK

- Author jimgilmour1 05-Jul-2010 1700 GMT

(C) Copyright Nokia 2014. All rights reserved.