qt static , mingwm10.dll and deployment under windows environment

After spending countless hours creating functions, algorithms and design, You application finally is done. And what now? At this point important decision must be made. Do I deploy it using Qt dynamic or static library? Under Window environment it's unusual to redistribute application with "many dll's" (at least with many Qt libs). In that case linking it dynamically is pointless. Another choice is to build, and link it against, Qt statically. In some cases it can be really pain in an …

To build Qt statically you just need to exec these commands:

```
cd %qtdir%
configure -static -[other option]
make sub-src
```

but that won't resolve all problems. If "MinGW" is used as compiler there is this 'nasty' "mingwm10.dll" that need to be distributed with application also. How to get rid of that? Simply edit mkspecs. Click "START" -> "Run", and type :

```
notepad %qtdir%\mkspecs\win32-g++\qmake.conf
```

or simply find that file. Then edit line

```
QMAKE_LFLAGS        = -enable-stdcall-fixup -Wl,-enable-auto-import -Wl,-enable-runtime-pseudo-reloc
```

and add (bold red "-static"):

```
QMAKE_LFLAGS        = -static -enable-stdcall-fixup -Wl,-enable-auto-import -Wl,-enable-runtime-pseudo-reloc
```

that will get rid of mingwm10.dll problem. After that issue these commands

```
cd %qtdir%
configure -static -release -no-exceptions -[other parameters like -mmx -sse -sse2 -3dnow etc...]
make sub-src [here you can make some coffee etc.. it will take some time] [1]
```

LUGRU

Some random thoughts about subject of my interest.

GRAPHICS

HARDWARE

PROGRAMING

ARCHIVES

July 2009 (1)
March 2009 (4)
February 2009 (1)

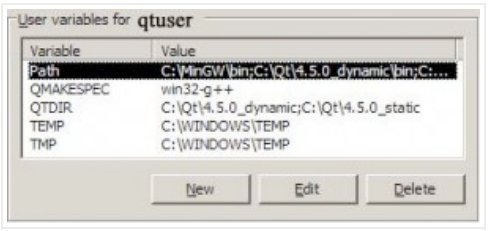
I also want to point out that without "-no-exceptions" parameter you application will still "want" that mingwm10.dll [I learnt that hard way ☹].

My tip here is to delete everything in "%qtdir%\demos" and %qtdir%\examples" so they wont be recompiled.

At this point everything is done. Application is compiled using static library, with means only "one file" need to be redistributed… but.

How to test it out? Of course you can send it to a friend, but that's too much of an effort. Very few people have two computers at home, so that solution also is out of question. Is there a simple way? Answer is yes 😀 .

First of all, what we need to do is take care of global variables. Right click on *"My computer"* then *"Properties" ->"Advanced" -> "Environment Variables"*. In window, that will pop out, you can see two lists. One is called "User variables for %you user name%" and second one "System Variables". Move any "Qt" and "MinGW" related entries to "User variables …" (see below picture) and "Delete" them from "System variables". That way only when ***You*** are logged in, Qt related variables will apply.



After that create another user in system. Call it for example "test" or whatever you like and reboot PC. Next time log-in as You normally do. Go to where application is located and right click on it. Chose this time "Run as…", then click on option "The following user: " and chose user that you created before. Type password etc. and run application. At this point everything should work fine, because program is compiled statically and don't depend on any dll's.

For example if application was compiled with Qt dynamic, debug "mode", you will see "This application has failed to start because mingwm10.dll was not found. …", but when app was compiled using Qt statical with release "mode" everything should work fine.

Side note here, to test if You correctly set variables use debug compiled program. First program should complain about mingwm10.dll then about missing Qt debug dll's, exactly in that order (as mentioned earlier for Qt dynamic, debug).

Another thing that should be mentioned is BIG size of statical compiled app's. It's because everything, even unneeded classes/functions within in example "QtCore.dll" are linked. In that case You can do somethings to take care of this. First run "strip" [2] to strip all unneeded symbols:

```
strip -s [other parameters] myApplication.exe
```

From my experience it won't help much, but every KB counts. Usually what i do next is compressing my programs with amazing tool called "UPX" [3]. It's "Ultimate Packer for eXecutables" and allows to compress content of any executable. I add to my variables, "Path", location to "UPX" so i don't need to copy/paste any files that way. In command line type:

```
cd %path_to_my_app%
upx -9 -[any other parameters you like] my_application.exe
```

Here is my little "shrinking" test. Test app (OpenGL test GUI app, with some labels etc.) that I used for this required these modules: "QtCore", "GtGui", "GtOpenGL". After compiling it with static Qt 4.5.0 size of my "exe" was:

```
original = 8 393 216 bytes [8,00 MB]
strip -s   =  8 393 216 bytes (that means no additional symbols were linked
```

> *into a app – so static release compilation of Qt works fine) [8,00 MB]*
> *upx -9 -compress-icons=3 = 3 037 696 bytes [2,89 MB]*

In the end 2.89 MB, big but not as much as 8.00 MB. There is also another way to shrink files, but that include messing with Qt src files and reconfiguring whole Qt.

I hope, as usual, that this article will be of any help to someone and sorry for my bad “enrglish”, my skills in that field are “little” rusty and all … but i try my best here ^^’. Also I want to state that if at some point I’m wrong about something let me know and I will gladly correct my errors.

This article applies to MinGW compiler.

Reference links:

[1] http://wiki.qtcentre.org/index.php?title=Building_static_Qt_on_Windows
[2] http://upx.sourceforge.net/
[3] http://www.mingw.org/wiki/Large_executables

Add new tag, deployment, environment, mingwm10.dll, qt static, strip, windows

This entry was posted on Tuesday, March 24th, 2009, 06:20 and is filed under Programing. You can follow any responses to this entry through RSS 2.0. Both comments and pings are currently closed.

Comments are closed.