Qt on Apple Silicon

Monday June 21, 2021 by Tor Arne Vestbø (https://www.qt.io/blog/author/tor-arne-vestbø) | Comments

When Apple announced the macOS transition to `arm64` last year with their new Apple Silicon M1 chip, we immediately started prototyping native support in Qt — initially on developer transition kits (DTK), and later on production hardware once that became available. The Rosetta translation layer already took care of running existing Qt applications on Apple Silicon, but we wanted native `arm64` builds, to squeeze out all the power from this new chipset.

Luckily for us, Qt already had good cross compilation support, as well as `arm64` support thanks to our iOS port, so bringing Qt up on Apple Silicon didn't initially take too much effort. The devil was in the details though.

One major hurdle was convincing the build system to not only treat `arm64` as a supported configuration on macOS, but to allow building Qt for both x86_64 and arm64 in one go, producing so called universal builds.

Another was ensuing that all our third party dependencies such as Chromium, PCRE, and OpenSSL were available and updated with `arm64`-support.

And, last but not least, we needed to add `arm64` macOS into our CI so we could run all the tests, which due to lack of virtualization options required some rethinking and additional work.

I'm happy to say that these issues have been resolved, and Qt on Apple Silicon support is scheduled for the upcoming Qt 6.2 release. This includes both cross compiling to `arm64`, as well as developing Qt applications on Apple Silicon.

Trying it out

You can try out Qt on Apple Silicon already now by installing the 6.2 preview from the Qt installer. The Qt SDK is fully universal, and should run on both Intel and Apple Silicon hardware.

You can also check out the 6.2 branch of the Qt repositories and build Qt from source. By default, Qt will be built for your local architecture, i.e. `x86_64` if you're on an Intel Mac, and `arm64` if you're on an Apple Silicon Mac. To produce a universal build, add the following argument to configure:

```
./configure <other options> -- -DCMAKE_OSX_ARCHITECTURES=
```

Once you have a Qt build for Apple Silicon your can use CMake like normal to configure and build your application. CMake also defaults to building for your local architecture, so to produce a universal build of your application, add the same argument when configuring:

```
cmake ~/src/myapp -DCMAKE_OSX_ARCHITECTURES="x86_64;arm64
```

Note that if you have third party dependencies these must be built as universal binaries as well.

For more details about building Qt applications for Apple Silicon see the documentation (https://doc-snapshots.qt.io/qt6-6.2/macos.html#architectures).

Building and debugging in Qt Creator

As Qt Creator is still built as a non-universal binary, it will default to producing `x86_64` binaries, regardless of which architecture your machine is.

To build for `arm64`, add `-DCMAKE_OSX_ARCHITECTURES=arm64` explicitly to the "Initial CMake parameters" of the project build settings, or `QMAKE_APPLE_DEVICE_ARCHS=arm64` to the qmake "Additional arguments" field.

Note: to attach a debugger you will need an `x86_64` build of your application.

These issues will be fixed before the final 6.2 release. Please let us know (https://bugreports.qt.io/secure/CreateIssue!default.jspa) if you find other issues.

What about Qt 5?

Our initial focus was on getting Qt for Apple Silicon into shape for Qt 6, but once 6.2 is out we'll take a look at the Qt 5.15 commercial LTS, and how we can improve the situation there as well.

Some patches already went into Qt 5, so Qt 5.15.4 and above should build and run by passing `-device-option QMAKE_APPLE_DEVICE_ARCHS=arm64` to configure. Note that this configuration is not tested in CI, and is hence unsupported at this point.

Applications built for x86_64 will run through the Rosetta translation layer, which should in most cases work fine. We are aware of a few bugs there, but please do not hesitate to file more (https://bugreports.qt.io/secure/CreateIssue!default.jspa) if you find any issues 😊

Share with your friends

(http://www.facebook.com/sharer.php?u=https://www.qt.io/blog/qt-on-apple-silicon, https://twitter.com/share?url=https://www.qt.io/blog/qt-on-apple-silicon&text=Qt%20on%20Apple%20Silicon, https://www.linkedin.com/shareArticle?mini=true&url=https://www.qt.io/blog/qt-on-apple-silicon, mailto:?subject=Qt%20on%20Apple%20Silicon&body=https://www.qt.io/blog/qt-on-apple-silicon)

Blog Topics:

Dev Loop (https://www.qt.io/blog/tag/loop)

macOS (https://www.qt.io/blog/tag/macos)

Apple (https://www.qt.io/blog/tag/apple)

Comments

Login

Add a comment

M↓ MARKDOWN ADD COMMENT

K **Kelteseth**
1 point · 7 months ago

Nice! Can you explicitly add the ARM support to the macOS Component description? It only states "Qt 6.2.0 Prebuilt Components for macOS". This makes it confusing because on Windows the ARM support section is split into two components, with x64 and ARM.

 **Tor Arne Vestbø (https://github.com/torarnv)**
1 point · 7 months ago

Good idea! I don't know why the Windows ARM support is split into two, but in any case it makes sense to mention explicitly that the macOS package has support for both architectures.

M **Martin Bříza**
1 point · 7 months ago

When is QtCreator coming to ARM64? The Rosetta build keeps crashing on me, about twice a day and since it's never been a problem for me anywhere else, I suspect the architecture is the culprit.

 **Tor Arne Vestbø (https://github.com/torarnv)**
0 points · 7 months ago

Qt Creator will be arm64 at a later point (post 6.2), since it's still built against Qt 5.15. Please file an issue (https://bugreports.qt.io/secure/CreateIssue!default.jspa) about the crash you are seeing, so that we can investigate it further, thanks!

Adrian Carpenter (https://github.com/fizzyade)
1 point · 7 months ago

Does it handle universal binaries yet?

I had Qt 5 running on apple arm64 the day my DTK arrived, and I ended up creating a tool (makeuniversal) that took at Qt build and produced a universal version, the same tool is used on my own binaries to create universal ones?

Tor Arne Vestbø (https://github.com/torarnv)
1 point · 7 months ago

Yes, you can build a universal binary of your application by adding `-DCMAKE_OSX_ARCHITECTURES="x86_64;arm64"` to the CMake arguments, as described in the blog post.

J

James Lomic
0 points · 7 months ago

I Need QtCreator on Android =) !

Tor Arne Vestbø (https://github.com/torarnv)
0 points · 7 months ago

Sorry, can't help you there :)

C

cliff
0 points · 7 months ago

Does the -device-option QMAKE*APPLE*DEVICE_ARCHS=arm64 work for qmake, both Qt5.15 and Qt6? Frankly, I don't like CMake and don't want to learn CMake

Tor Arne Vestbø (https://github.com/torarnv)
0 points · 7 months ago

If you are talking about building Qt 6, it's no longer build with qmake, it's built with CMake, so `CMAKE_OSX_ARCHITECTURES` is the way to affect the architecture there.

If you're talking about using a Qt 6 build with a qmake based user project, then `QMAKE_APPLE_DEVICE_ARCHS` should work.

If you're talking about Qt 5, then in principle `QMAKE_APPLE_DEVICE_ARCHS` is the way to both build Qt and to set the architecture for user projects, but as Qt 5 support for Apple Silicon is not yet officially supported I can't promise that it works 100%.

C

cliff
0 points · 7 months ago

> As Qt5.15.5 or higher is only for Commercial users, so if it can fully work for Apple M1, there will be a good contribution for Qt Company. Being courage!

Panagiotis Kanavos
0 points · 7 months ago

Hello,

I'm trying to build my project for arm64 but no matter what I try, Qt Creator and CMake always switch to x86. I'm including `-DCMAKE_OSX_ARCHITECTURES=arm64` in the "Initial CMake parameters" and also trying with switches in my CMakeLists.txt file, but I always get x86. Any hints? Using Qt 6.2.0 alpha and Qt Creator 4.15.0 rc1 from the online installer.

Tor Arne Vestbø (https://github.com/torarnv)
0 points · 7 months ago

Are you on an M1 Mac, or are you cross-compiling on an Intel Mac? As Qt Creator is still an x86_64 binary the former has some sharp edges still. https://codereview.qt-project.org/c/qt-creator/qt-creator/+/356033 (https://codereview.qt-project.org/c/qt-creator/qt-creator/+/356033) should take care of that.

Does it work on the command line, if you run CMake outside of Qt Creator? If not, please file an issue and I'll have a look.

If you're using qmake you may want to cherry-pick https://codereview.qt-project.org/c/qt/qtbase/+/355970 (https://codereview.qt-project.org/c/qt/qtbase/+/355970) onto your 6.2 tree (until it's been included in the next online installer snapshot).

Panagiotis Kanavos
1 point · 7 months ago

Thanks for the reply! Yes, I'm using an M1 Mac and trying to build only for arm64.

My project is quite large with many submodules, so I just tried and built a subproject with CMake outside Qt Creator successfully. Inside Qt Creator I'm having problems with OpenMP at the moment so I can't move on. Once I solve this, I'll update you.

Thank you!

Panagiotis Kanavos
0 points · 7 months ago

> > Indeed, the problem is Qt Creator. I can compile the project outside Qt Creator, from the command line or XCode. I guess I'll have to do without Qt Creator on my M1 Mac until it ships for arm64.

O **Ole André V. Ravnås**
0 points · 7 months ago

If anybody else needs a bandaid for building Qt 5.x for Apple Silicon, this hacky patch works for me:
https://github.com/frida/cryptoshark/blob/master/tools/macos/qtbase-apple-silicon.patch
(https://github.com/frida/cryptoshark/blob/master/tools/macos/qtbase-apple-silicon.patch)

This is the build recipe I'm using:
https://github.com/frida/cryptoshark/blob/master/tools/macos/build-qt
(https://github.com/frida/cryptoshark/blob/master/tools/macos/build-qt)

Read Next

(https://www.qt.io/blog/qt-6.2-alpha-released?hsLang=en)

(https://www.qt.io/blog/qt-6.2-beta-released?hsLang=en)

(https://www.qt.io/blog/qt-6.2-vs.-qt-5.15-the-feature-parity-comparison?hsLang=en)

Jun 23, 2021

Qt 6 Reaches Feature Parity with Qt 5 - the Qt 6.2 Alpha Released

Read Article
(https://www.qt.io/blog/qt-6.2-alpha-released?hsLang=en)

Jul 6, 2021

Qt 6.2 Beta Released

Read Article
(https://www.qt.io/blog/qt-6.2-beta-released?hsLang=en)

Sep 2, 2021

Qt 6.2 vs. Qt 5.15 – The Feature Parity Comparison

Read Article
(https://www.qt.io/blog/qt-6.2-vs.-qt-5.15-the-feature-parity-comparison?hsLang=en)