aqtinstall Documentation

Release 3.1.13

Hiroshi Miura

Apr 13, 2024

CONTENTS

1 Installation **3**
- 1.1 Requirements . . . 3
- 1.2 Install by pip command . . . 3

2 Command changes **5**

3 Getting Started **7**
- 3.1 Installing Qt . . . 7
- 3.2 External 7-zip extractor . . . 8
- 3.3 Changing the output directory . . . 9
- 3.4 Installing Modules . . . 9
- 3.5 Installing Qt for Android . . . 10
- 3.6 Installing Qt for WASM . . . 11
- 3.7 Installing Docs . . . 12
- 3.8 Installing Tools . . . 12
- 3.9 Installing a subset of Qt archives [Advanced] . . . 14

4 Command Line Options **19**
- 4.1 Generic commands . . . 19
- 4.2 List-* Commands . . . 19
- 4.3 Install-* Commands . . . 24
- 4.4 Legacy subcommands . . . 29

5 Command examples **31**

6 Configuration **35**
- 6.1 Settings . . . 36

7 Contribution guide **39**
- 7.1 Resources . . . 39
- 7.2 Bug triage . . . 39
- 7.3 Send patch . . . 39
- 7.4 CI tests . . . 39

8 Contributor Covenant Code of Conduct **41**
- 8.1 Our Pledge . . . 41
- 8.2 Our Standards . . . 41
- 8.3 Enforcement Responsibilities . . . 42
- 8.4 Scope . . . 42
- 8.5 Enforcement . . . 42
- 8.6 Enforcement Guidelines . . . 42

8.7 Attribution . . . 43

9 Security Policy **45**
9.1 Supported Versions . . . 45
9.2 Reporting a Vulnerability . . . 45

10 Authors **47**

11 ChangeLog **49**
11.1 Unreleased . . . 49
11.2 v3.1.13 (13, Apr. 2024) . . . 49
11.3 v3.1.12 (2, Mar. 2024) . . . 49
11.4 v3.1.11 (28, Nov. 2023) . . . 50
11.5 v3.1.10 (14, Nov. 2023) . . . 50
11.6 v3.1.9 (6, Nov. 2023) . . . 51
11.7 v3.1.8 (1, Nov. 2023) . . . 51
11.8 v3.1.7 (1, Aug. 2023) . . . 52
11.9 v3.1.6 (4, May, 2023) . . . 52
11.10 v3.1.5 (30, Mar. 2023) . . . 52
11.11 v3.1.4 (25, Mar. 2023) . . . 53
11.12 v3.1.3 (2, Mar. 2023) . . . 53
11.13 v3.1.2 (17, Feb. 2023) . . . 53
11.14 v3.1.1 (10, Feb. 2023) . . . 53
11.15 v3.1.0 (5, Dec. 2022) . . . 54
11.16 v3.0.2 (26, Oct. 2022) . . . 54
11.17 v3.0.1 (30, Sep. 2022) . . . 54
11.18 v3.0.0 (29, Sep. 2022) . . . 55
11.19 v2.2.3 (17, Aug. 2022) . . . 55
11.20 v2.2.2 (11, Aug. 2022) . . . 55
11.21 v2.2.0 (2, Aug. 2022) . . . 56
11.22 v2.1.0 (14, Apr. 2022) . . . 56

12 Changes until v2.0.6 **59**
12.1 v2.0.6 (7, Feb. 2022) . . . 59
12.2 v2.0.5 (11, Dec. 2021) . . . 59
12.3 v2.0.4 (5, Dec. 2021) . . . 59
12.4 Fixed . . . 59
12.5 Changed . . . 60
12.6 v2.0.3 (25, Nov. 2021) . . . 60
12.7 v2.0.2 (1, Nov. 2021) . . . 60
12.8 v2.0.1 (29, Oct. 2021) . . . 61
12.9 v2.0.0 (29, Sep. 2021) . . . 61
12.10 v1.2.5 (14, Aug. 2021) . . . 63
12.11 v1.2.4 (17, Jul. 2021) . . . 63
12.12 v1.2.3 (14, Jul. 2021) . . . 63
12.13 v1.2.2 (1, Jul. 2021) . . . 64
12.14 v1.2.1 (22, Jun. 2021) . . . 64
12.15 v1.2.0 (21, Jun. 2021) . . . 64
12.16 v1.1.6 (2, May. 2021) . . . 65
12.17 v1.1.5 (8, Apr. 2021) . . . 65
12.18 v1.1.4 (2, Apr. 2021) . . . 66
12.19 v1.1.3 (26, Feb. 2021) . . . 66
12.20 v1.1.2 (20, Feb. 2021) . . . 66
12.21 v1.1.1 (13, Feb. 2021) . . . 66
12.22 v1.1.0 (12, Feb. 2021) . . . 66

12.23 v1.0.0 (4, Feb. 2021) . . . 67
12.24 v0.11.1 (21, Jan. 2021) . . . 67
12.25 v0.11.0 (21, Jan. 2021) . . . 68
12.26 v0.10.1 (11, Dec. 2020) . . . 68
12.27 v0.10.0 (25, Nov. 2020) . . . 68
12.28 v0.9.8 (4, Nov. 2020) . . . 69
12.29 v0.9.7 (4, Oct. 2020) . . . 69
12.30 v0.9.6 (7, Sep. 2020) . . . 69
12.31 v0.9.5 (18, Aug. 2020) . . . 69
12.32 v0.9.4 (2, Aug. 2020) . . . 69
12.33 v0.9.3 (1, Aug. 2020) . . . 70
12.34 v0.9.2 (19, June. 2020) . . . 70
12.35 v0.9.1 (13, June. 2020) . . . 70
12.36 v0.9.0 (31, May. 2020) . . . 70
12.37 v0.9.0b3 (21, May. 2020) . . . 71
12.38 v0.9.0b2 (21, May. 2020) . . . 71
12.39 v0.9.0b1 (10, May. 2020) . . . 71
12.40 v0.8 (26, Mar. 2020) . . . 72
12.41 v0.8b1 (12, Mar. 2020) . . . 72
12.42 v0.8a4 (6, Mar., 2020) . . . 72
12.43 v0.8a3 (5, Mar., 2020) . . . 72
12.44 v0.8a1 (28, Feb., 2020) . . . 73
12.45 v0.7.4 (15, Feb., 2020) . . . 73
12.46 v0.7.3 (14, Feb., 2020) . . . 73
12.47 v0.7.2 (11, Feb., 2020) . . . 74
12.48 v0.7.1 (13, Jan., 2020) . . . 74
12.49 v0.7 (13, Jan., 2020) . . . 74
12.50 v0.7b1 (10, Jan., 2020) . . . 74
12.51 v0.7a2 (7, Jan., 2020) . . . 74
12.52 v0.7a1 (29, Nov., 2019) . . . 75
12.53 v0.6b1 (23, Nov., 2019) . . . 75
12.54 v0.6a2 (19, Nov., 2019) . . . 75
12.55 v0.6a1 (17, Nov., 2019) . . . 76
12.56 v0.5 (10, Nov., 2019) . . . 76
12.57 v0.5b2 (8, Oct., 2019) . . . 77
12.58 v0.5b1 (8, Oct., 2019) . . . 77
12.59 v0.4.3 (25, Sep, 2019) . . . 77
12.60 v0.4.2 (28, Jul, 2019) . . . 77
12.61 v0.4.1 (01, Jun, 2019) . . . 78
12.62 v0.4.0 (29, May, 2019) . . . 78
12.63 v0.3.1 (15, March, 2019) . . . 79
12.64 v0.3.0 (8, March, 2019) . . . 79
12.65 v0.2.0 (7, March, 2019) . . . 80
12.66 v0.1.0 (5, March, 2019) . . . 80
12.67 v0.0.2 (4, March, 2019) . . . 80
12.68 Added . . . 80
12.69 Changed . . . 80
12.70 Fixed . . . 81
12.71 v0.0.1 (2, March, 2019) . . . 81

13 Indices and tables **83**

Index **85**

Contents:

CHAPTER

ONE

INSTALLATION

1.1 Requirements

- Minimum Python version: 3.7.5
- Dependencies: requests, py7zr, semantic_version, patch, texttable, bs4, defusedxml, humanize

1.2 Install by pip command

Same as usual, it can be installed with `pip`

```
$ pip install aqtinstall
```

CHAPTER

TWO

COMMAND CHANGES

From version 2.0.0, sub commands are changed. The previous versions of these sub commands have been retained for backwards compatibility, but are no longer recommended.

New sub commands	Legacy sub commands	Note
install-qt	install	Version moved after target
install-tool	tool	Arguments are changed New syntax doesn't take version
install-example	examples	Version moved after target Caution with last (s)
install-src	src	Version moved after target New command only can take –kde option
install-doc	doc	Version moved after target
	list	Legacy list commands are removed.
list-qt		
list-tool		

CHAPTER

THREE

GETTING STARTED

`aqt` is a tool that can be used to install Qt, modules, tools related to Qt, source, docs, and examples, available at https://download.qt.io/. Before running `aqt`, you will need to tell `aqt` exactly what you want it to install. This section of the documentation is meant to walk you through the process of finding out what packages are available to `aqt`, so you can tell `aqt` what you want to install.

Please note that every `aqt` subcommand has a `--help` option; please use it if you are having trouble!

3.1 Installing Qt

General usage of `aqt` looks like this:

```
aqt install-qt <host> <target> (<Qt version> | <spec>) [<arch>]
```

If you have installed `aqt` with pip, you can run it with the command script `aqt`, but in some cases you may need to run it as `python -m aqt`. Some older operating systems may require you to specify Python version 3, like this: `python3 -m aqt`.

To use `aqt` to install Qt, you will need to tell `aqt` four things:

1. The host operating system (windows, mac, linux, or linux_arm64)
2. The target SDK (desktop, android, ios, or winrt)
3. The version of Qt you would like to install
4. The target architecture

Keep in mind that Qt for IOS is only available on Mac OS, and Qt for WinRT is only available on Windows.

In current versions of Qt, mac binaries are universal. As of Qt 6.7.0, Linux desktop now supports the arm64 architecture. This is implemented as a new host type - `linux` is amd64, `linux_arm64` is arm64.

To find out what versions of Qt are available, you can use the *aqt list-qt command*. This command will print all versions of Qt available for Windows Desktop:

```
$ aqt list-qt windows desktop
5.9.0 5.9.1 5.9.2 5.9.3 5.9.4 5.9.5 5.9.6 5.9.7 5.9.8 5.9.9
5.10.0 5.10.1
5.11.0 5.11.1 5.11.2 5.11.3
5.12.0 5.12.1 5.12.2 5.12.3 5.12.4 5.12.5 5.12.6 5.12.7 5.12.8 5.12.9 5.12.10 5.12.11
5.13.0 5.13.1 5.13.2
5.14.0 5.14.1 5.14.2
5.15.0 5.15.1 5.15.2
```

(continues on next page)

(continued from previous page)

```
6.0.0 6.0.1 6.0.2 6.0.3 6.0.4
6.1.0 6.1.1 6.1.2
6.2.0
```

Notice that the version numbers are sorted, grouped by minor version number, and separated by a single space-character. The output of all of the *aqt list-qt* commands is intended to make it easier for you to write programs that consume the output of *aqt list-qt*.

Because the *aqt list-qt* command directly queries the Qt downloads repository at https://download.qt.io/, the results of this command will always be accurate. The Available Qt versions wiki page was last modified at some point in the past, so it may or may not be up to date.

Now that we know what versions of Qt are available, let's choose version 6.2.0.

The next thing we need to do is find out what architectures are available for Qt 6.2.0 for Windows Desktop. To do this, we can use *aqt list-qt* with the `--arch` flag:

```
$ aqt list-qt windows desktop --arch 6.2.0
win64_mingw81 win64_msvc2019_64 win64_msvc2019_arm64 wasm_32
```

Notice that this is a very small subset of the architectures listed in the Available Qt versions wiki page. If we need to use some architecture that is not on this list, we can use the Available Qt versions wiki page to get a rough idea of what versions support the architecture we want, and then use *aqt list-qt* to confirm that the architecture is available.

Let's say that we want to install Qt 6.2.0 with architecture `win64_mingw81`. The installation command we need is:

```
$ aqt install-qt windows desktop 6.2.0 win64_mingw81
```

Let's say that we want to install the next version of Qt 6.2 as soon as it is available. We can do this by using a SimpleSpec instead of an explicit version:

```
$ aqt install-qt windows desktop 6.2 win64_mingw81
```

As of Qt 6.7.0, arm64 architecture is now supported for linux desktop. It is implemented using both a different host (`linux_arm64`) and architecture (`linux_gcc_arm64`).

```
$ aqt install-qt linux_arm64 desktop 6.7.0 linux_gcc_arm64
```

3.2 External 7-zip extractor

By default, `aqt` extracts the 7zip archives stored in the Qt repository using py7zr, which is installed alongside `aqt`. You can specify an alternate 7zip command path instead by using the `-E` or `--external` flag. For example, you could use 7-zip on a Windows desktop, using this command:

```
C:\> aqt install-qt windows desktop 6.2.0 gcc_64 --external 7za.exe
```

On Linux, you can specify p7zip, a Linux port of 7-zip, which is often installed by default, using this command:

```
$ aqt install-qt linux desktop 6.2.0 gcc_64 --external 7z
```

3.3 Changing the output directory

By default, `aqt` will install all of the Qt packages into the current working directory, in the subdirectory `./<Qt version>/<arch>/`. For example, if we install Qt 6.2.0 for Windows desktop with arch `win64_mingw81`, it would end up in `./6.2.0/win64_mingw81`.

If you would prefer to install it to another location, you will need to use the `-O` or `--outputdir` flag. This option also works for all of the other subcommands that begin with `aqt install-`.

To install to `C:\Qt`, the default directory used by the standard gui installer, you may use this command:

```
C:\> mkdir Qt
C:\> aqt install-qt --outputdir c:\Qt windows desktop 6.2.0 win64_mingw81
```

3.4 Installing Modules

Let's say we need to install some modules for Qt 5.15.2 on Windows Desktop. First we need to find out what the modules are called, and we can do that with *aqt list-qt* with the `--modules` flag. Each version of Qt has a different list of modules for each host OS/ target SDK/ architecture combination, so we will need to supply *aqt list-qt* with that information:

```
$ aqt list-qt windows desktop --modules 5.15.2 win64_mingw81
qtcharts qtdatavis3d qtlottie qtnetworkauth qtpurchasing qtquick3d
qtquicktimeline qtscript qtvirtualkeyboard qtwebengine qtwebglplugin
```

Let's say that we want to know more about these modules before we install them. We can use the `--long-modules` flag for that:

```
$ aqt list-qt windows desktop --long-modules 5.15.2 win64_mingw81
   Module Name                          Display Name
========================================================================
debug_info          Desktop MinGW 8.1.0 64-bit Debug Information Files
qtcharts            Qt Charts for MinGW 8.1.0 64-bit
qtdatavis3d         Qt Data Visualization for MinGW 8.1.0 64-bit
qtlottie            Qt Lottie Animation for MinGW 8.1.0 64-bit
qtnetworkauth       Qt Network Authorization for MinGW 8.1.0 64-bit
qtpurchasing        Qt Purchasing for MinGW 8.1.0 64-bit
qtquick3d           Qt Quick 3D for MinGW 8.1.0 64-bit
qtquicktimeline     Qt Quick Timeline for MinGW 8.1.0 64-bit
qtscript            Qt Script for MinGW 8.1.0 64-bit
qtvirtualkeyboard   Qt Virtual Keyboard for MinGW 8.1.0 64-bit
qtwebglplugin       Qt WebGL Streaming Plugin for MinGW 8.1.0 64-bit
```

Note that if your terminal is wider than 95 characters, this command will show release dates and sizes in extra columns to the right. If you try this, you will notice that `debug_info` is 5.9 gigabytes installed.

Also, notice that the 'Display Name' indicates which compiler the module is intended to be used with. In this case, for the architecture `win64_mingw81`, you will most likely want to use the "MinGW 8.1.0 64-bit" compiler. Here's what the command prints when you use it with the ambiguously-named `win64_mingw` architecture:

```
$ python -m aqt list-qt windows desktop --long-modules 6.2.4 win64_mingw
   Module Name                          Display Name
========================================================================
```

(continues on next page)

(continued from previous page)

```
debug_info          Desktop MinGW 11.2.0 64-bit debug information files
qt3d                Qt 3D for MinGW 11.2.0 64-bit
qt5compat           Qt 5 Compatibility Module for MinGW 11.2.0 64-bit
qtactiveqt          Qt 3D for MinGW 11.2.0 64-bit
qtcharts            Qt Charts for MinGW 11.2.0 64-bit
...
```

You can find out how to install MinGW 8.1.0 and 11.2.0 in the *Installing Tools* section.

Let's say that we want to install `qtcharts` and `qtnetworkauth`. We can do that by using the `-m` flag with the *aqt install-qt* command. This flag receives the name of at least one module as an argument:

```
$ aqt install-qt windows desktop 5.15.2 win64_mingw81 -m qtcharts qtnetworkauth
```

If we wish to install all the modules that are available, we can do that with the `all` keyword:

```
$ aqt install-qt windows desktop 5.15.2 win64_mingw81 -m all
```

Remember that the *aqt list-qt* command is meant to be scriptable? One way to install all modules available for Qt 5.15.2 is to send the output of *aqt list-qt* into *aqt install-qt*, like this:

```
$ aqt install-qt windows desktop 5.15.2 win64_mingw81 \
      -m $(aqt list-qt windows desktop --modules 5.15.2 win64_mingw81)
```

You will need a Unix-style shell to run this command, or at least git-bash on Windows. The `xargs` equivalent to this command is an exercise left to the reader.

If you want to install all available modules, you are probably better off using the `all` keyword, as discussed above. This scripting example is presented to give you a sense of how to accomplish something more complicated. Perhaps you want to install all modules except `qtnetworkauth`; you could write a script that removes `qtnetworkauth` from the output of *aqt list-qt*, and pipe that into *aqt install-qt*. For example:

```
$ aqt install-qt windows desktop 5.15.2 win64_mingw81 \
    -m $(for mod in $(aqt list-qt windows desktop --modules 5.15.2 win64_mingw81); \
    do [[ "$mod" != "qtnetworkauth" ]] && echo -n "$mod "; done)
```

3.5 Installing Qt for Android

Let's install Qt for Android. This will be similar to installing Qt for Desktop on Windows.

Note: Versions of aqtinstall older than 3.1.0 required the use of the `--extensions` and `--extension` flag to list any architectures, modules, or archives for Qt 6 and above. These flags are no longer necessary, so please do not use them.

```
$ aqt list-qt windows android                          # Print Qt versions available
5.9.0 5.9.1 ...
...
6.4.0

$ aqt list-qt windows android --arch 6.2.4             # Print architectures available
android_x86_64 android_armv7 android_x86 android_arm64_v8a
```

(continues on next page)

(continued from previous page)

```
$ aqt list-qt windows android --modules 6.2.4 android_armv7   # Print modules available
qt3d qt5compat qtcharts qtconnectivity qtdatavis3d ...

$ aqt install-qt windows android 6.2.4 android_armv7 -m qtcharts qtnetworkauth   #␣
↪Install
```

Please note that when you install Qt6 for android or ios, the installation will not be functional unless you install the corresponding desktop version of Qt alongside it. You can do this automatically with the `--autodesktop` flag:

```
$ aqt install-qt linux android 6.2.4 android_armv7 -m qtcharts qtnetworkauth --
↪autodesktop
```

3.6 Installing Qt for WASM

To find out how to install Qt for WASM, we will need to use the `wasm_32` architecture. We can find out whether or not that architecture is available for our version of Qt with the `--arch` flag.

```
$ python -m aqt list-qt windows desktop --arch 6.1.3
win64_mingw81 win64_msvc2019_64
$ python -m aqt list-qt windows desktop --arch 6.2.0
win64_mingw81 win64_msvc2019_64 win64_msvc2019_arm64 wasm_32
```

Not every version of Qt supports WASM. This command shows us that we cannot use WASM with Qt 6.1.3.

Please note that the WASM architecture for Qt 6.5.0+ changed from `wasm_32` to `wasm_singlethread` and `wasm_multithread`. Always use `aqt list-qt` to check what architectures are available for the desired version of Qt.

We can check the modules available as before:

```
$ aqt list-qt windows desktop --modules 5.15.2 wasm_32   # available modules
qtcharts qtdatavis3d qtlottie qtnetworkauth qtpurchasing qtquicktimeline qtscript
qtvirtualkeyboard qtwebglplugin
```

We can install Qt for WASM as before:

```
$ aqt install-qt windows desktop 5.15.2 wasm_32 -m qtcharts qtnetworkauth
```

Please note that when you install Qt for WASM version 6 and above, the installation will not be functional unless you install a non-WASM desktop version of Qt alongside it. You can do this automatically with the `--autodesktop` flag:

```
$ aqt install-qt linux desktop 6.2.0 wasm_32 -m qtcharts qtnetworkauth --autodesktop
```

3.7 Installing Docs

The *aqt list-doc* command lists documentation archives for a given Qt version:

```
$ aqt list-doc mac 6.6.1
qdoc qmake qt5 qtassistant qtcmake qtconcurrent qtcore qtdbus qtdesigner
qtdistancefieldgenerator qtdoc qtgui qthelp qtlabsplatform qtlinguist qtnetwork
qtopengl qtplatformintegration qtprintsupport qtqml qtqmlcore qtqmlmodels qtqmltest
qtqmlworkerscript qtqmlxmllistmodel qtquick qtquickcontrols qtquickdialogs qtsql
qtsvg qttestlib qtuitools qtwaylandcompositor qtwidgets qtxml
```

All of the above archives will be installed when you run `aqt install-doc mac 6.6.1` without any other flags or arguments. You can select a subset of them with the `--archives` flag.

The `--modules` flag lists additional documentation modules that can be installed:

```
$ aqt list-doc mac 6.6.1 --modules
qt3d qt5compat qtactiveqt qtbluetooth qtcharts qtdatavis3d qtgraphs qtgrpc qthttpserver
qtimageformats qtlocation qtlottie qtmultimedia qtnetworkauth qtnfc qtpdf qtpositioning
qtquick3d qtquick3dphysics qtquickeffectmaker qtquicktimeline qtremoteobjects qtscxml
qtsensors qtserialbus qtserialport qtshadertools qtspeech qtvirtualkeyboard qtwebchannel
qtwebengine qtwebsockets qtwebview
```

The `--archives` and `--modules` flags can be used together. For example, to only install the docs for `qtquick` and `qt3d`, use the *aqt install-doc* command like this:

```
$ aqt install-doc mac 6.6.1 --archives qtquick --modules qt3d
INFO    : Downloading qt3d...
INFO    : Downloading qtquick...
INFO    : Redirected: qt.mirror.constant.com
INFO    : Redirected: qt.mirror.constant.com
INFO    : Finished installation of qt3d-documentation.tar.xz in 1.51932292
INFO    : Finished installation of qtquick-documentation.tar.xz in 2.63531679
INFO    : Finished installation
INFO    : Time elapsed: 4.00115146 second
```

3.8 Installing Tools

Let's find out what tools are available for Windows Desktop by using the *aqt list-tool* command:

```
$ aqt list-tool windows desktop
tools_vcredist
...
tools_qtcreator
tools_qt3dstudio
tools_openssl_x86
tools_openssl_x64
tools_openssl_src
tools_ninja
tools_mingw
tools_mingw90
tools_ifw
```

(continues on next page)

(continued from previous page)

```
tools_conan
tools_cmake
```

Let's see what tool variants are available in `tools_mingw`:

```
$ aqt list-tool windows desktop tools_mingw
qt.tools.mingw47
qt.tools.win32_mingw48
qt.tools.win32_mingw482
qt.tools.win32_mingw491
qt.tools.win32_mingw492
qt.tools.win32_mingw530
qt.tools.win32_mingw730
qt.tools.win32_mingw810
qt.tools.win64_mingw730
qt.tools.win64_mingw810
```

This gives us a list of things that we could install using *aqt install-tool*. Let's see some more details, using the `-l` or `--long` flag:

```
$ aqt list-tool windows desktop tools_mingw -l

   Tool Variant Name            Version          Release Date
============================================================
qt.tools.mingw47          4.7.2-1-1              2013-07-01
qt.tools.win32_mingw48    4.8.0-1-1              2013-07-01
qt.tools.win32_mingw482   4.8.2                  2014-05-08
qt.tools.win32_mingw491   4.9.1-3                2016-05-31
qt.tools.win32_mingw492   4.9.2-1                2016-05-31
qt.tools.win32_mingw530   5.3.0-2                2017-04-27
qt.tools.win32_mingw730   7.3.0-1-202004170606   2020-04-17
qt.tools.win32_mingw810   8.1.0-1-202004170606   2020-04-17
qt.tools.win64_mingw730   7.3.0-1-202004170606   2020-04-17
qt.tools.win64_mingw810   8.1.0-1-202004170606   2020-04-17
```

The `-l` flag causes *aqt list-tool* to print a table that shows plenty of data pertinent to each tool variant available in `tools_mingw`. *aqt list-tool* additionally prints the 'Display Name' and 'Description' for each tool if your terminal is wider than 95 characters; terminals that are narrower than this cannot display this table in a readable way.

Please be aware that the tool `tools_mingw90` appears to be mislabelled:

```
$ aqt list-tool windows desktop tools_mingw90 -l

   Tool Variant Name            Version          Release Date
============================================================
qt.tools.win64_mingw900   9.0.0-1-202203221220   2022-03-22

$ aqt list-tool windows desktop tools_mingw90 -l

   Tool Variant Name            Version          Release Date      Display Name          ␣
↪  Description
=============================================================================================================
qt.tools.win64_mingw900   9.0.0-1-202203221220   2022-03-22     MinGW 11.2.0 64-bit   ␣
```

(continues on next page)

(continued from previous page)

```
↪MinGW-builds 11.2.0                                                    64-
↪bit toolchain with                                                     ␣
↪gcc 11.2.0
```

The 'narrow display' for `tools_mingw90` cuts off the two columns of the table that show you what's really in that package: `MinGW 11.2.0 64-bit`. If you are using the `win64_mingw` architecture for Qt 6.2.2+, then this is probably the compiler you want to install (see *long_modules explanation*).

Now let's install `mingw`, using the *aqt install-tool* command. This command receives four parameters:

1. The host operating system (windows, mac, linux, or linux_arm64)
2. The target SDK (desktop, android, ios, or winrt)
3. The name of the tool (this is `tools_mingw` in our case)
4. (Optional) The tool variant name. We saw a list of these when we ran *aqt list-tool* with the `tool name` argument filled in.

To install `mingw`, you could use this command (please don't):

```
$ aqt install-tool windows desktop tools_mingw    # please don't run this!
```

Using this command will install every tool variant available in `tools_mingw`; in this case, you would install 10 different versions of the same tool. For some tools, like `qtcreator` or `ifw`, this is an appropriate thing to do, since each tool variant is a different program. However, for tools like `mingw` and `vcredist`, it would make more sense to use *aqt list-tool* to see what tool variants are available, and then install just the tool variant you are interested in, like this:

```
$ aqt install-tool windows desktop tools_mingw qt.tools.win64_mingw730
```

Please note that `aqt install-tool` does not recognize the `installscript.qs` related to each tool. When you install these tools with the standard gui installer, the installer may use the `installscript.qs` script to make additional changes to your system. If you need those changes to occur, it will be your responsibility to make those changes happen, because `aqt` is not capable of running this script.

3.9 Installing a subset of Qt archives [Advanced]

3.9.1 Introduction

You may have noticed that by default, `aqt install-qt` installs a lot of archives that you may or may not need, and a typical installation can take up more disk space than necessary. If you installed the module `debug_info`, it may have installed more than 1 gigabyte of data. This section will help you to reduce the footprint of your Qt installation.

Note: Be careful about using the `--archives` flag; it is marked `Advanced` for a reason! It is very easy to misuse this command and end up with a Qt installation that is missing the components that you need. Don't use it unless you know what you are doing!

3.9.2 Minimum Qt Installation

Normally, when you run `aqt install-qt`, the program will print a long list of archives that it is downloading, extracting, and installing, including `qtbase`, `qtmultimedia`, `qt3d`, and ~25 more items. We can use the `--archives` flag to choose which of these archives we will actually install. The `--archives` flag can only affect two modules: the base Qt installation and the `debug_info` module.

Note: In this documentation, **"modules"**, **"archives"**, and **"the base Qt installation"** refer to different things, and are defined here:

- **Archives**: In this context, an **archive** is a bundle of files compressed with the 7zip algorithm. It exists on a disk drive as a file with the extension `.7z`.
- **Modules**: The Qt repository organizes groups of archives into modules. A **module** contains one or more **archives**.
- **the base Qt installation**: By definition, this is just another **module** that contains 20-30 **archives**. This documentation refers to it as **the base Qt installation** instead of a **module** for several reasons:
  - The `aqt install-qt` installs this module by default.
  - You cannot specify this module with `aqt install-qt --modules`.
  - The `aqt list-qt --modules` command is incapable of printing this module.
  - `aqt` transforms the names of modules as they exist in the Qt repository so that they are easier to read and write. If the name of **the base Qt installation** were transformed using the same rules, the name would be empty.

    The fully-qualified name of the **base Qt installation** module is usually something like `qt.qt6.620.gcc_64`. The fully-qualified name of the `qtcharts` module could be something like `qt.qt6.620.qtcharts.gcc_64`. It would be difficult to read and write a list of 20 modules with the prefix `qt.qt6.620.` and the suffix `.gcc_64`, because these parts are repetitive and not meaningful. Only the `qtcharts` part is useful.

Let's say that we want to install Qt 5.15.2 for Linux desktop, using the gcc_64 architecture. The `qtbase` archive includes the bare minimum for a working Qt installation, and we can install it alone with the `--archives` flag:

```
$ aqt install-qt linux desktop 5.15.2 --archives qtbase
```

This time, `aqt install-qt` will only install one archive, `qtbase`, instead of the ~27 archives it installs by default.

3.9.3 Installing More Than The Bare Minimum

Let's say that the `qtbase` archive is missing some features that you need. Using the `--archives qtbase` flag causes `aqt install-qt` to omit roughly 27 archives. We can print a list of these archives with the `aqt list-qt --archives` command:

```
$ aqt list-qt linux desktop --archives 5.15.2 gcc_64
icu qt3d qtbase qtconnectivity qtdeclarative qtgamepad qtgraphicaleffects qtimageformats
qtlocation qtmultimedia qtquickcontrols qtquickcontrols2 qtremoteobjects qtscxml
qtsensors qtserialbus qtserialport qtspeech qtsvg qttools qttranslations qtwayland
qtwebchannel qtwebsockets qtwebview qtx11extras qtxmlpatterns
```

Here, we have used the `--archives` flag with two arguments: the version of Qt we are interested in, and the architecture we are using. As a result, the command printed a list of archives that are part of the base (non-minimal) Qt installation.

Let's say we need to use `qtmultimedia`, `qtdeclarative`, `qtsvg`, and nothing else. Remember that the `qtbase` archive is required for a minimal working Qt installation. We can install these archives using this command:

```
$ aqt install-qt linux desktop 5.15.2 --archives qtbase qtmultimedia qtdeclarative qtsvg
```

3.9.4 Installing Modules With Archives Specified

As of aqt v2.1.0, the `--archives` flag will only apply to the base Qt installation and to the `debug_info` module. Previous versions of aqt required that when installing modules with the `--archives` flag, the user must specify archives for each module, otherwise they would not be installed. This behavior has been changed to prevent such mistakes.

Let's say that we need to install the bare minimum Qt 5.15.2, with the modules `qtcharts` and `qtlottie`:

```
$ aqt install-qt linux desktop 5.15.2 --modules qtcharts qtlottie --archives qtbase
```

This command will successfully install 3 archives: 1 for `qtbase`, and one each for the two modules. If we had tried to use this command with previous versions of aqt, we would not have installed the two modules because we did not specify them in the `--archives` list.

Note: You can still misuse the `--archives` flag by omitting the `qtbase` archive, or by omitting archives that another archive or module is dependent on. You may not notice that there is a problem until you try to compile a program, and compilation fails.

3.9.5 Installing the `debug_info` module

Now let's say we need to install the `debug_info` module, which is particularly large: around one gigabyte. We do not want to install all of it, so we can use `aqt install-qt --archives` to choose which archives we want to install. Remember that the `--archives` flag

`aqt list-qt --archives` to print which archives are part of the `debug_info` module:

```
$ aqt list-qt linux desktop --archives 5.15.2 gcc_64 debug_info
qt3d qtbase qtcharts qtconnectivity qtdatavis3d qtdeclarative qtgamepad␣
↪qtgraphicaleffects
qtimageformats qtlocation qtlottie qtmultimedia qtnetworkauth qtpurchasing qtquick3d
qtquickcontrols qtquickcontrols2 qtquicktimeline qtremoteobjects qtscript qtscxml␣
↪qtsensors
qtserialbus qtserialport qtspeech qtsvg qttools qtvirtualkeyboard qtwayland qtwebchannel
qtwebengine qtwebglplugin qtwebsockets qtwebview qtx11extras qtxmlpatterns
```

This is a lot of archives. Note that there's a name collision between the `debug_info` archives and the archives in every other module/Qt base install: this is because there's a `debug_info` archive that corresponds to almost every other archive available.

Let's install Qt with `qtcharts` and `debug_info` with some archives specified:

```
$ aqt install-qt linux desktop --modules qtcharts debug_info \
                               --archives qtcharts qtbase qtdeclarative
```

Notice what we did here: We specified the `qtcharts` and `debug_info` modules, and we specified the `qtbase`, `qtcharts`, and `qtdeclarative` archives. This will install a total of 6 archives:

- the 3 archives named `qtbase`, `qtcharts`, and `qtdeclarative` from the `debug_info` module,

- the 1 archive `qtcharts` from the `qtcharts` module, and
- the 2 archives `qtbase` and `qtdeclarative` from the base Qt installation.

Note: At present, `aqt install-qt` is incapable of installing any archive from the `debug_info` module without also installing the corresponding module from the base Qt installation. For instance, you cannot install the `debug_info` archive for `qtbase` without also installing the usual `qtbase` archive.

CHAPTER

FOUR

COMMAND LINE OPTIONS

The CLI uses argparse to parse the command line options so the short or long versions may be used and the long options may be truncated to the shortest unambiguous abbreviation.

4.1 Generic commands

```
aqt help
```

show generic help

```
aqt version
```

display version

4.2 List-* Commands

These commands are used to list the packages available for installation with `aqt`.

4.2.1 list-qt command

```
aqt list-qt [-h | --help]
            [-c | --config]
            [--spec <specification>]
            [--modules      (<Qt version> | latest) <architecture> |
             --long-modules (<Qt version> | latest) <architecture> |
             --arch         (<Qt version> | latest) |
             --archives     (<Qt version> | latest) architecture [modules...]
             --latest-version]
            <host> [<target>]
```

List available versions of Qt, targets, modules, and architectures.

host
: linux, linux_arm64, windows or mac

target
: desktop, winrt, ios or android. When omitted, the command prints all the targets available for a host OS. Note that winrt is only available on Windows, and ios is only available on Mac OS.

--help, -h

Display help text

--spec <Specification>

Print versions of Qt within a SimpleSpec that specifies a range of versions. You can specify partial versions, inequalities, etc. "*" would match all versions of Qt; ">6.0.2,<6.2.0" would match all versions of Qt between 6.0.2 and 6.2.0, etc. For example, `aqt list-qt windows desktop --spec "5.12"` would print all versions of Qt for Windows Desktop beginning with 5.12. May be combined with any other flag to filter the output of that flag.

--modules (<Qt version> | latest) <architecture>

This flag lists all the modules available for Qt 5.X.Y with a host/target/architecture combination, or the latest version of Qt if `latest` is specified. You can list available architectures by using `aqt list-qt` with the `--arch` flag described below.

--long-modules (<Qt version> | latest) <architecture>

Long display for modules: Similar to `--modules`, but shows extra metadata associated with each module. This metadata is displayed in a table that includes long display names for each module. If your terminal is wider than 95 characters, `aqt list-qt` will also display release dates and sizes for each module. An example of this output is displayed below.

```
$ python -m aqt list-qt windows desktop --long-modules latest win64_mingw

   Module Name                         Display Name                       Release Date  ␣
↪Download Size   Installed Size
==========================================================================================================
debug_info          Desktop MinGW 11.2.0 64-bit debug information files   2022-07-07    ␣
↪1.0G            6.4G
qt3d                Qt 3D for MinGW 11.2.0 64-bit                         2022-07-07    ␣
↪2.8M            21.3M
qt5compat           Qt 5 Compatibility Module for MinGW 11.2.0 64-bit     2022-07-07    ␣
↪679.3K          2.5M
qtactiveqt          Qt 3D for MinGW 11.2.0 64-bit                         2022-07-07    ␣
↪5.9M            32.6M
qtcharts            Qt Charts for MinGW 11.2.0 64-bit                     2022-07-07    ␣
↪713.0K          7.5M
qtconnectivity      Qt Connectivity for MinGW 11.2.0 64-bit               2022-07-07    ␣
↪227.5K          1.5M
qtdatavis3d         Qt Data Visualization for MinGW 11.2.0 64-bit         2022-07-07    ␣
↪565.7K          4.3M
qthttpserver        Qt HTTP Server for MinGW 11.2.0 64-bit                2022-07-07    ␣
↪73.2K           372.6K
qtimageformats      Qt Image Formats for MinGW 11.2.0 64-bit              2022-07-07    ␣
↪184.6K          705.5K
qtlanguageserver    Qt language Server for MinGW 11.2.0 64-bit            2022-07-07    ␣
↪300.1K          1.8M
qtlottie            Qt Lottie Animation for MinGW 11.2.0 64-bit           2022-07-07    ␣
↪131.7K          704.0K
qtmultimedia        Qt Multimedia for MinGW 11.2.0 64-bit                 2022-07-07    ␣
↪9.7M            79.2M
qtnetworkauth       Qt Network Authorization for MinGW 11.2.0 64-bit      2022-07-07    ␣
↪85.5K           507.6K
qtpositioning       Qt Positioning for MinGW 11.2.0 64-bit                2022-07-07    ␣
↪347.2K          2.2M
```

(continues on next page)

(continued from previous page)

```
qtquick3d           Qt Quick 3D for MinGW 11.2.0 64-bit                2022-07-07  ␣
↪13.0M           75.4M
qtquick3dphysics    Quick: 3D Physics for MinGW 11.2.0 64-bit          2022-07-07  ␣
↪35.5M           203.9M
qtquicktimeline     Qt Quick Timeline for MinGW 11.2.0 64-bit          2022-07-07  ␣
↪54.6K           301.4K
qtremoteobjects     Qt Remote Objects for MinGW 11.2.0 64-bit          2022-07-07  ␣
↪424.4K          2.0M
qtscxml             Qt State Machine for MinGW 11.2.0 64-bit           2022-07-07  ␣
↪448.5K          2.9M
qtsensors           Qt Sensors for MinGW 11.2.0 64-bit                 2022-07-07  ␣
↪175.7K          2.0M
qtserialbus         Qt SerialBus for MinGW 11.2.0 64-bit               2022-07-07  ␣
↪208.8K          1.2M
qtserialport        Qt SerialPort for MinGW 11.2.0 64-bit              2022-07-07  ␣
↪58.3K           255.3K
qtshadertools       Qt Shader Tools for MinGW 11.2.0 64-bit            2022-07-07  ␣
↪1.2M            4.1M
qtspeech            Qt Speech for MinGW 11.2.0 64-bit                  2022-07-07  ␣
↪81.8K           427.9K
qtvirtualkeyboard   Qt Virtual Keyboard for MinGW 11.2.0 64-bit        2022-07-07  ␣
↪2.1M            6.0M
qtwebchannel        Qt WebChannel for MinGW 11.2.0 64-bit              2022-07-07  ␣
↪114.0K          500.3K
qtwebsockets        Qt WebSockets for MinGW 11.2.0 64-bit              2022-07-07  ␣
↪96.3K           509.6K
qtwebview           Qt WebView for MinGW 11.2.0 64-bit                 2022-07-07  ␣
↪64.2K           470.7K
```

--arch (<Qt version> | latest)

Qt version in the format of "5.X.Y". When set, this prints all architectures available for Qt 5.X.Y with a host/target, or the latest version of Qt if `latest` is specified.

--archives (<Qt version> | latest) architecture [modules...]

This flag requires a list of at least two arguments: 'Qt version' and 'architecture'. The 'Qt version' argument can be in the format "5.X.Y" or the "latest" keyword. You can use the `--arch` flag to see a list of acceptable values for the 'architecture' argument. Any following arguments must be the names of modules available for the preceding version and architecture. You can use the `--modules` flag to see a list of acceptable values.

If you do not add a list of modules to this flag, this command will print a list of all the archives that make up the base Qt installation.

If you add a list of modules to this flag, this command will print a list of all the archives that make up the specified modules.

The purpose of this command is to show you what arguments you can pass to the *archives flag* when using the `install-*` commands. This flag allows you to avoid installing parts of Qt that you do not need.

--latest-version

Print only the newest version available May be combined with the `--spec` flag.

4.2.2 list-src command

```
aqt list-src [-h | --help]
             [-c | --config]
             <host> (<Qt version> | <spec>)
```

List source archives available for installation using the *install-src command*.

host

linux, linux_arm64, windows or mac

Qt version

This is a Qt version such as 5.9.7, 5.12.1 etc. Use the *list-qt command* to list available versions.

spec

This is a SimpleSpec that specifies a range of versions. If you type something in the `<Qt version>` positional argument that cannot be interpreted as a version, it will be interpreted as a SimpleSpec, and `aqt` will select the highest available version within that SimpleSpec.

For example, `aqt list-src mac 5.12` would print archives for the latest version of Qt 5.12 available (5.12.11 at the time of this writing).

4.2.3 list-doc command

```
aqt list-doc [-h | --help]
             [-c | --config]
             [-m | --modules]
             <host> (<Qt version> | <spec>)
```

List documentation archives and modules available for installation using the *install-doc command*.

By default, `list-doc` will print a list of archives available for installation using the *install-doc command*, with the `--archives` option.

host

linux, linux_arm64, windows or mac

Qt version

This is a Qt version such as 5.9.7, 5.12.1 etc. Use the *list-qt command* to list available versions.

spec

This is a SimpleSpec that specifies a range of versions. If you type something in the `<Qt version>` positional argument that cannot be interpreted as a version, it will be interpreted as a SimpleSpec, and `aqt` will select the highest available version within that SimpleSpec.

For example, `aqt list-doc mac 5.12` would print archives for the latest version of Qt 5.12 available (5.12.11 at the time of this writing).

--modules

This flag causes `list-doc` to print a list of modules available for installation using the *install-doc command*, with the `--modules` option.

4.2.4 list-example command

```
aqt list-example [-h | --help]
                 [-c | --config]
                 [-m | --modules]
                 <host> (<Qt version> | <spec>)
```

List example archives and modules available for installation using the *install-example command*.

By default, `list-example` will print a list of archives available for installation using the *install-example command*, with the `--archives` option.

host

linux, linux_arm64, windows or mac

Qt version

This is a Qt version such as 5.9.7, 5.12.1 etc. Use the *list-qt command* to list available versions.

spec

This is a SimpleSpec that specifies a range of versions. If you type something in the `<Qt version>` positional argument that cannot be interpreted as a version, it will be interpreted as a SimpleSpec, and `aqt` will select the highest available version within that SimpleSpec.

For example, `aqt list-example mac 5.12` would print archives for the latest version of Qt 5.12 available (5.12.11 at the time of this writing).

--modules

This flag causes `list-example` to print a list of modules available for installation using the *install-example command*, with the `--modules` option.

4.2.5 list-tool command

```
aqt list-tool [-h | --help] [-c | --config] [-l | --long] <host> [<target>] [<tool name>]
```

List available tools

host

linux, linux_arm64, windows or mac

target

desktop, winrt, ios or android. When omitted, the command prints all the targets available for a host OS. Note that winrt is only available on Windows, and ios is only available on Mac OS.

tool name

The name of a tool. Use `aqt list-tool <host> <target>` to see accepted values. When set, this prints all 'tool variant names' available.

The output of this command is meant to be used with the *aqt install-tool* below.

--help, -h

Display help text

--long, -l

Long display: shows extra metadata associated with each tool variant. This metadata is displayed in a table, and includes versions and release dates for each tool. If your terminal is wider than 95 characters, `aqt list-tool` will also display the names and descriptions for each tool. An example of this output is displayed below.

```
$ python -m aqt list-tool windows desktop tools_conan -l

 Tool Variant Name           Version         Release Date     Display Name            ␣
↪Description
==========================================================================================
qt.tools.conan         1.33-202102101246     2021-02-10     Conan 1.33          Conan␣
↪command line tool 1.33
qt.tools.conan.cmake   0.16.0-202102101246   2021-02-10     Conan conan.cmake   Conan␣
↪conan.cmake (0.16.0)
```

4.3 Install-* Commands

These commands are used to install Qt, tools, source, docs, and examples.

4.3.1 Common Options

Most of these commands share the same command line options, and these options are described here:

--help, -h

Display help text

--outputdir, -O <Output Directory>

Specify output directory. By default, aqt installs to the current working directory.

--base, -b <base url>

Specify mirror site base url such as -b `https://mirrors.dotsrc.org/qtproject` where 'online' folder exist.

--config, -c <settings_file_path>

Specify the path to your own `settings.ini` file. See *the Configuration section*.

--timeout <timeout(sec)>

The connection timeout, in seconds, for the download site. (default: 5 sec)

--external, -E <7zip command>

Specify external 7zip command path. By default, aqt uses py7zr for this task.

In the past, our users have had success using 7-zip on Windows, Linux and Mac. You can install 7-zip on Windows with Choco. The Linux/Mac port of 7-zip is called `p7zip`, and you can install it with brew on Mac, or on Linux with your package manager.

--internal

Use the internal extractor, py7zr

--keep, -k

Keep downloaded archive when specified, otherwise remove after install. Use `--archive-dest` <path> to choose where aqt will place these files. If you do not specify a download destination, aqt will place these files in the current working directory.

--archive-dest <path>

Set the destination path for downloaded archives (temp directory by default). All downloaded archives will be automatically deleted unless you have specified the `--keep` option above, or `aqt` crashes.

Note that this option refers to the intermediate `.7z` archives that `aqt` downloads and then extracts to `--outputdir`. Most users will not need to keep these files.

--modules, -m (<list of modules> | all)

Specify extra modules to install as a list. Use the appropriate `aqt list-*` command to list available modules:

Install command	List command	Usage of list command
install-qt	*list-qt command*	`list-qt <host> <target> --modules <version> <arch>`
install-example	*list-example command*	`list-example <host> <version> --modules`
install-doc	*list-doc command*	`list-doc <host> <version> --modules`

This option only applicable to `install-qt`, `install-example`, and `install-doc`.

You can install multiple modules like this:

```
$ aqt install-* <host> <target> <Qt version> -m qtcharts qtdatavis3d qtlottie␣
↪qtnetworkauth \
    qtpurchasing qtquicktimeline qtscript qtvirtualkeyboard qtwebglplugin
```

If you wish to install every module available, you may use the `all` keyword instead of a list of modules, like this:

```
aqt install-* <host> <target> <Qt version> <arch> -m all
```

--archives <list of archives>

[Advanced] Specify subset of archives to **limit** installed archives. It will only affect the base Qt installation and the `debug_info` module. This is advanced option and not recommended to use for general usage. Main purpose is speed up CI/CD process by limiting installed modules. It can cause broken installation of Qt SDK.

This option is applicable to all the `install-*` commands except for `install-tool`.

You can print a list of all acceptable values to use with this command by using the appropriate `aqt list-*` command:

Install command	List command	Usage of list command
install-qt	*list-qt command*	`list-qt <host> <target> --archives <version>`
install-example	*list-example command*	`list-example <host> <version>`
install-src	*list-src command*	`list-src <host> <version>`
install-doc	*list-doc command*	`list-doc <host> <version>`

4.3.2 install-qt command

```
aqt install-qt
    [-h | --help]
    [-c | --config]
    [-O | --outputdir <directory>]
    [-b | --base <mirror url>]
    [--timeout <timeout(sec)>]
    [-E | --external <7zip command>]
    [--internal]
    [-k | --keep]
    [-d | --archive-dest] <path>
    [-m | --modules (all | <module> [<module>...])]
    [--archives <archive> [<archive>...]]
```

(continues on next page)

(continued from previous page)

```
    [--autodesktop]
    [--noarchives]
    <host> <target> (<Qt version> | <spec>) [<arch>]
```

Install Qt library, with specified version and target. There are various combinations to accept according to Qt version.

host

linux, linux_arm64, windows or mac. The operating system on which the Qt development tools will run.

target

desktop, ios, winrt, or android. The type of device for which you are developing Qt programs. If your target is ios, please be aware that versions of Qt older than 6.2.4 are expected to be non-functional with current versions of XCode (applies to any XCode greater than or equal to 13).

Qt version

This is a Qt version such as 5.9.7, 5.12.1 etc. Use the *list-qt command* to list available versions.

spec

This is a SimpleSpec that specifies a range of versions. If you type something in the `<Qt version>` positional argument that cannot be interpreted as a version, it will be interpreted as a SimpleSpec, and `aqt` will select the highest available version within that SimpleSpec.

For example, `aqt install-qt mac desktop 5.12` would install the newest version of Qt 5.12 available, and `aqt install-qt mac desktop "*"` would install the highest version of Qt available.

When using this option, `aqt` will print the version that it has installed in the logs so that you can verify it easily.

arch

The compiler architecture for which you are developing. Options:

- gcc_64 or linux_gcc_64 for linux desktop
- linux_gcc_arm64 for linux_arm64 desktop
- clang_64 for mac desktop
- win64_msvc2019_64, win64_msvc2017_64, win64_msvc2015_64, win32_msvc2015, win32_mingw53 for windows desktop
- android_armv7, android_arm64_v8a, android_x86, android_x86_64 for android

  Use the *list-qt command* to list available architectures.

--autodesktop

If you are installing an ios, android, WASM, or msvc_arm64 version of Qt6, the corresponding desktop version of Qt must be installed alongside of it. Turn this option on to install it automatically. This option will have no effect if the desktop version of Qt is not required.

--noarchives

[Advanced] Specify not to install all base packages. This is advanced option and you should use it with `--modules` option. This allow you to add modules to existent Qt installation.

See *common options*.

4.3.3 install-src command

```
aqt install-src
    [-h | --help]
    [-c | --config]
    [-O | --outputdir <directory>]
    [-b | --base <mirror url>]
    [--timeout <timeout(sec)>]
    [-E | --external <7zip command>]
    [--internal]
    [-k | --keep]
    [-d | --archive-dest] <path>
    [--archives <archive> [<archive>...]]
    [--kde]
    <host> [<target>] (<Qt version> | <spec>)
```

Install Qt source code for the specified version and target.

host

linux, linux_arm64, windows or mac

target

Deprecated and marked for removal in a future version of aqt. This parameter exists for backwards compatibility reasons, and its value is ignored.

Qt version

This is a Qt version such as 5.9.7, 5.12.1 etc. Use the *list-qt command* to list available versions.

spec

This is a SimpleSpec that specifies a range of versions. If you type something in the `<Qt version>` positional argument that cannot be interpreted as a version, it will be interpreted as a SimpleSpec, and `aqt` will select the highest available version within that SimpleSpec.

For example, `aqt install-src mac 5.12` would install sources for the newest version of Qt 5.12 available, and `aqt install-src mac "*"` would install sources for the highest version of Qt available.

--kde

by adding `--kde` option, KDE patch collection is applied for qtbase tree. It is only applied to Qt 5.15.2. When specified version is other than it, command will abort with error when using `--kde`.

See *common options*.

4.3.4 install-doc command

```
aqt install-doc
    [-h | --help]
    [-c | --config]
    [-O | --outputdir <directory>]
    [-b | --base <mirror url>]
    [--timeout <timeout(sec)>]
    [-E | --external <7zip command>]
    [--internal]
    [-k | --keep]
    [-d | --archive-dest] <path>
```

(continues on next page)

(continued from previous page)

```
    [-m | --modules (all | <module> [<module>...])]
    [--archives <archive> [<archive>...]]
    <host> [<target>] (<Qt version> | <spec>)
```

Install Qt documentation for the specified version and target.

host

linux, linux_arm64, windows or mac

target

Deprecated and marked for removal in a future version of aqt. This parameter exists for backwards compatibility reasons, and its value is ignored.

Qt version

This is a Qt version such as 5.9.7, 5.12.1 etc. Use the *list-qt command* to list available versions.

spec

This is a SimpleSpec that specifies a range of versions. If you type something in the `<Qt version>` positional argument that cannot be interpreted as a version, it will be interpreted as a SimpleSpec, and `aqt` will select the highest available version within that SimpleSpec.

For example, `aqt install-doc mac 5.12` would install documentation for the newest version of Qt 5.12 available, and `aqt install-doc mac "*"` would install documentation for the highest version of Qt available.

See *common options*.

4.3.5 install-example command

```
aqt install-example
    [-h | --help]
    [-c | --config]
    [-O | --outputdir <directory>]
    [-b | --base <mirror url>]
    [--timeout <timeout(sec)>]
    [-E | --external <7zip command>]
    [--internal]
    [-k | --keep]
    [-d | --archive-dest] <path>
    [-m | --modules (all | <module> [<module>...])]
    [--archives <archive> [<archive>...]]
    <host> [<target>] (<Qt version> | <spec>)
```

Install Qt examples for the specified version and target.

host

linux, linux_arm64, windows or mac

target

Deprecated and marked for removal in a future version of aqt. This parameter exists for backwards compatibility reasons, and its value is ignored.

Qt version

This is a Qt version such as 5.9.7, 5.12.1 etc. Use the *list-qt command* to list available versions.

spec

This is a SimpleSpec that specifies a range of versions. If you type something in the `<Qt version>` positional argument that cannot be interpreted as a version, it will be interpreted as a SimpleSpec, and `aqt` will select the highest available version within that SimpleSpec.

For example, `aqt install-example mac 5.12` would install examples for the newest version of Qt 5.12 available, and `aqt install-example mac "*"` would install examples for the highest version of Qt available.

See *common options*.

4.3.6 install-tool command

```
aqt install-tool
    [-h | --help]
    [-c | --config]
    [-O | --outputdir <directory>]
    [-b | --base <mirror url>]
    [--timeout <timeout(sec)>]
    [-E | --external <7zip command>]
    [--internal]
    [-k | --keep]
    [-d | --archive-dest] <path>
    <host> <target> <tool name> [<tool variant name>]
```

Install tools like QtIFW, mingw, Cmake, Conan, and vcredist.

host

linux, linux_arm64, windows or mac

target

desktop, ios or android

tool name

install tools specified. tool name may be 'tools_openssl_x64', 'tools_vcredist', 'tools_ninja', 'tools_ifw', 'tools_cmake'

tool variant name

Optional field to specify tool variant. It may be required for vcredist and mingw installation. tool variant names may be 'qt.tools.win64_mingw810', 'qt.tools.vcredist_msvc2013_x64'.

You should use the *list-tool command* to display what tools and tool variant names are available.

See *common options*.

4.4 Legacy subcommands

The subcommands `install`, `tool`, `src`, `doc`, and `examples` have been deprecated in favor of the newer `install-*` commands, but they remain in aqt in case you still need to use them. Documentation for these older commands is still available at https://aqtinstall.readthedocs.io/en/v1.2.4/

CHAPTER

FIVE

COMMAND EXAMPLES

Example: Installing Qt SDK 5.12.12 for Linux with QtCharts and QtNetworkAuth:

```
pip install aqtinstall
sudo aqt install-qt --outputdir /opt linux desktop 5.12.12 -m qtcharts qtnetworkauth
```

Example: Installing the newest LTS version of Qt 5.12:

```
pip install aqtinstall
sudo aqt install-qt linux desktop 5.12 gcc_64
```

Example: Installing the newest LTS version of Qt 6.7 for linux arm64:

```
pip install aqtinstall
sudo aqt install-qt linux_arm64 desktop 6.7
```

Example: Installing Android (armv7) Qt 5.13.2:

```
aqt install-qt linux android 5.13.2 android_armv7
```

Example: Installing Android (armv7) Qt 6.4.2:

```
aqt install-qt linux android 6.4.2 android_armv7 --autodesktop
```

Example: Install examples, doc and source:

```
aqt install-example windows 5.15.2 -m qtcharts qtnetworkauth
aqt install-doc windows 5.15.2 -m qtcharts qtnetworkauth
aqt install-src windows 5.15.2 --archives qtbase --kde
```

Example: Print archives available for installation with `install-example/doc/src`:

```
aqt list-example windows 5.15.2
aqt list-doc windows 5.15.2
aqt list-src windows 5.15.2
```

Example: Print modules available for installation with `install-example/doc`:

```
aqt list-example windows 5.15.2 --modules
aqt list-doc windows 5.15.2 --modules
```

Example: Install Web Assembly

```
aqt install-qt linux desktop 5.15.0 wasm_32
```

Example: Install Qt6 for Web Assembly

```
aqt install-qt linux desktop 6.2.4 wasm_32 --autodesktop
```

Example: List available versions of Qt on Linux

```
aqt list-qt linux desktop
```

Example: List available versions of Qt6 on macOS

```
aqt list-qt mac desktop --spec "6"
```

Example: List available modules for latest version of Qt on macOS

```
aqt list-qt mac desktop --modules latest clang_64   # prints 'qtquick3d qtshadertools',␣
↪etc
```

Example: List available architectures for Qt 6.1.2 on windows

```
aqt list-qt windows desktop --arch 6.1.2    # prints 'win64_mingw81 win64_msvc2019_64',␣
↪etc
```

Example: List available tools on windows

```
aqt list-tool windows desktop    # prints 'tools_ifw tools_qtcreator', etc
```

Example: List the variants of IFW available:

```
aqt list-tool linux desktop tools_ifw       # prints 'qt.tools.ifw.41'
# Alternate: `tools_` prefix is optional
aqt list-tool linux desktop ifw             # prints 'qt.tools.ifw.41'
```

Example: List the variants of IFW, including version, release date, description, etc.:

```
aqt list-tool linux desktop tools_ifw -l    # prints a table of metadata
```

Example: Install an Install FrameWork (IFW):

```
aqt install-tool linux desktop tools_ifw
```

Example: Install vcredist on Windows:

```
aqt install-tool windows desktop tools_vcredist
.\Qt\Tools\vcredist\vcredist_msvc2019_x64.exe /norestart /q
```

Example: Install MinGW 8.1.0 on Windows:

```
aqt install-tool -O c:\Qt windows desktop tools_mingw qt.tools.win64_mingw810
set PATH=C:\Qt\Tools\mingw810_64\bin
```

Example: Install MinGW 11.2.0 on Windows:

```
aqt install-tool -O c:\Qt windows desktop tools_mingw90
set PATH=C:\Qt\Tools\mingw1120_64\bin
```

Note: This is not a typo; it is a mislabelled tool name! `tools_mingw90` and the tool variant `qt.tools.win64_mingw900` do not contain MinGW 9.0.0; they actually contain MinGW 11.2.0! Verify with `aqt list-tool --long windows desktop tools_mingw90` in a wide terminal.

Example: Show help message

```
aqt help
```

CHAPTER

SIX

CONFIGURATION

`aqtinstall` can be configured through a configuration file. A default configuration is stored in `aqt/settings.ini` file.

You can specify custom configuration file through `AQT_CONFIG` environment variable or "-c" or "–config" command line option.

A file is like as follows:

```
[DEFAULTS]

[aqt]
concurrency: 4
baseurl: https://download.qt.io
7zcmd: 7z
print_stacktrace_on_error: False
always_keep_archives: False
archive_download_location: .
min_archive_size: 41

[requests]
connection_timeout: 3.5
response_timeout: 30
max_retries_on_connection_error: 5
retry_backoff: 0.1
max_retries_on_checksum_error: 5
max_retries_to_retrieve_hash: 5
hash_algorithm: sha256
INSECURE_NOT_FOR_PRODUCTION_ignore_hash: False

[mirrors]
trusted_mirrors:
    https://download.qt.io
blacklist:
    http://mirrors.ustc.edu.cn
    http://mirrors.tuna.tsinghua.edu.cn
    http://mirrors.geekpie.club
fallbacks:
    https://mirrors.ocf.berkeley.edu/qt
    https://ftp.jaist.ac.jp/pub/qtproject
    http://ftp1.nluug.nl/languages/qt
    https://mirrors.dotsrc.org/qtproject
```

(continues on next page)

(continued from previous page)

```
[kde_patches]
patches:
    0001-toolchain.prf-Use-vswhere-to-obtain-VS-installation-.patch
```

6.1 Settings

The `[aqt]` section configures basic behavior.

concurrency:
`concurrency` is a setting how many download concurrently starts. It should be a integer value.

baseurl:
`baseurl` is a URL of Qt download site. When you have your own Qt download site repository, you can set it here. It is as same as `--base` option.

7zcmd:
It is a command name of 7-zip. When `aqtinstall` is installed **without** recommended library `py7zr`, it is used to extract archive instead of `py7zr` library. When `--external` option specified, a value is override with option's one.

print_stacktrace_on_error:
`print_stacktrace_on_error` is either `True` or `False`. The `True` setting causes a stack trace to be printed to stderr any time an error occurs that will end the program. The `False` setting will hide the stack trace, unless an unhandled exception occurs.

always_keep_archives:
This is either `True` or `False`. The `True` setting turns on the `--keep` option every time you run aqt, and cannot be overridden by command line options. The `False` setting will require you to set `--keep` manually every time you run aqt, unless you don't want to keep `.7z` archives.

archive_download_location:
This is the relative or absolute path to the location in which `.7z` archives will be downloaded, when `--keep` is turned on. You can override this location with the `--archives-dest` option.

min_module_size:
This is the minimum decompressed size, in bytes, of the modules that aqt is permitted to list. The authors of aqt have discovered that the Qt repository contains a few mysteriously "empty" modules, including the examples modules for *qtlottie* and *qtquicktimeline*. These modules consist of a single archive that contains empty directories, and they are exactly 40 bytes when uncompressed. The authors feel that it is not useful for `aqt list-*` to list these empty modules. If you want to print these modules with `aqt list-*`, please feel free to change the `min_module_size` value to something less than 40.

This setting has no effect on your ability to install these modules. `aqt install-*` can will still install them without any warnings.

The `[requests]` section controls the way that `aqt` makes network requests.

connection_timeout:
`connection_timeout` is a timeout in second for connection. It is passed to `requests` library.

response_timeout:
`response_timeout` is a timeout in second how much time waiting for response. It is passed to `requests` library.

max_retries:
Deprecated; please do not use this setting. It has been replaced by the

max_retries_on_connection_error:
`max_retries_on_connection_error` is an integer that controls how many times `aqt` will try to reconnect to the server in the case of a connection error.

retry_backoff:
`retry_backoff` is a floating point number that controls how long `aqt` will sleep between failed connection attempts. Setting this value too low will hammer the server, and may result in no successful connections at all.

max_retries_on_checksum_error:
This setting controls how many times `aqt` will attempt to download a file, in the case of a checksum error.

hash_algorithm:
This is either `sha256`, `sha1` or `md5`. `sha256` is the only safe value to use here. Default is `sha256` if not set. See also `trusted_mirrors` setting.

INSECURE_NOT_FOR_PRODUCTION_ignore_hash:
This is either `True` or `False`. The `True` setting disables hash checking when downloading files. Although this is not recommended, this may help when hashes are not available. The `False` setting will enforce hash checking. This is highly recommended to avoid corrupted files.

The `[mirrors]` section is a configuration for mirror handling.

trusted_mirrors:
`trusted_mirrors` is a list of URLs that you trust to provide accurate checksums for all downloaded archives. This is a security feature; please do not change this value unless you know what you're doing!

`aqtinstall` downloads all checksums from mirrors in this list. These checksums are used to verify that every other file that `aqtinstall` downloads is, in fact, the correct file, and not a corrupt or malicious copy of the file. You may need to modify this list if the default mirrors are unreachable, or if you do not trust that they have not been compromised.

`aqtinstall` can safely download archive files from the fallback mirror list, and ensure that they are not malicious files, by checking them against the checksums downloaded from the `trusted_mirrors` list. `aqtinstall` uses the SHA-256 algorithm to perform this check.

blacklist:
It is a list of URL where is a problematic mirror site. Some mirror sites ignore a connection from IP addresses out of their preffered one. It will cause connection error or connection timeout. There are some known mirror sites in default. If you are not happy with the default sites, you can override them with custom settings.

fallbacks:
It is a list of URL where is a good for access. When mirror site cause an error, aqt use fallbacks when possible. You can find a list of mirrors at: https://download.qt.io/static/mirrorlist/

CHAPTER

SEVEN

CONTRIBUTION GUIDE

This is contribution guide for aqtinstall project. You are welcome to send a Pull-Request, reporting bugs and ask questions.

7.1 Resources

- Project owner: Hiroshi Miura
- Bug Tracker: Github issue Tracker
- Status: Beta
- Activity: moderate

7.2 Bug triage

Every report to github issue tracker should be in triage. whether it is bug, question or invalid.

7.3 Send patch

Here is small amount rule when you want to send patch the project;

1. every proposal for modification should send as 'Pull Request'
1. each pull request can consist of multiple commits.
1. you are encourage to split modifications to individual commits that are logical subpart.

7.4 CI tests

The project configured to use Azure Pipelines, Github actions and Coveralls for regression test. You can see test results on badge and see details in a web page linked from badge.

CHAPTER

EIGHT

CONTRIBUTOR COVENANT CODE OF CONDUCT

8.1 Our Pledge

We as members, contributors, and leaders pledge to make participation in our community a harassment-free experience for everyone, regardless of age, body size, visible or invisible disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, religion, or sexual identity and orientation.

We pledge to act and interact in ways that contribute to an open, welcoming, diverse, inclusive, and healthy community.

8.2 Our Standards

Examples of behavior that contributes to a positive environment for our community include:

- Demonstrating empathy and kindness toward other people
- Being respectful of differing opinions, viewpoints, and experiences
- Giving and gracefully accepting constructive feedback
- Accepting responsibility and apologizing to those affected by our mistakes, and learning from the experience
- Focusing on what is best not just for us as individuals, but for the overall community

Examples of unacceptable behavior include:

- The use of sexualized language or imagery, and sexual attention or advances of any kind
- Trolling, insulting or derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or email address, without their explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

8.3 Enforcement Responsibilities

Community leaders are responsible for clarifying and enforcing our standards of acceptable behavior and will take appropriate and fair corrective action in response to any behavior that they deem inappropriate, threatening, offensive, or harmful.

Community leaders have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, and will communicate reasons for moderation decisions when appropriate.

8.4 Scope

This Code of Conduct applies within all community spaces, and also applies when an individual is officially representing the community in public spaces. Examples of representing our community include using an official e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event.

8.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported to the community leaders responsible for enforcement at Hiroshi Miura <miurahr@linux.com>. All complaints will be reviewed and investigated promptly and fairly.

All community leaders are obligated to respect the privacy and security of the reporter of any incident.

8.6 Enforcement Guidelines

Community leaders will follow these Community Impact Guidelines in determining the consequences for any action they deem in violation of this Code of Conduct:

8.6.1 Phase 1. Correction

Community Impact:
Use of inappropriate language or other behavior deemed unprofessional or unwelcome in the community.

Consequence:
A private, written warning from community leaders, providing clarity around the nature of the violation and an explanation of why the behavior was inappropriate. A public apology may be requested.

8.6.2 Phase 2. Warning

Community Impact:
A violation through a single incident or series of actions.

Consequence:
A warning with consequences for continued behavior. No interaction with the people involved, including unsolicited interaction with those enforcing the Code of Conduct, for a specified period of time. This includes avoiding interactions in community spaces as well as external channels like social media. Violating these terms may lead to a temporary or permanent ban.

8.6.3 Phase 3. Temporary Ban

Community Impact:
A serious violation of community standards, including sustained inappropriate behavior.

Consequence:
A temporary ban from any sort of interaction or public communication with the community for a specified period of time. No public or private interaction with the people involved, including unsolicited interaction with those enforcing the Code of Conduct, is allowed during this period. Violating these terms may lead to a permanent ban.

8.6.4 Phase 4. Permanent Ban

Community Impact:
Demonstrating a pattern of violation of community standards, including sustained inappropriate behavior, harassment of an individual, or aggression toward or disparagement of classes of individuals.

Consequence:
A permanent ban from any sort of public interaction within the community.

8.7 Attribution

This Code of Conduct is adapted from the Contributor Covenant version 2.0. You can take it from Contributor Covenant homepage.

Community Impact Guidelines were inspired by Mozilla's code of conduct enforcement ladder.

For answers to common questions about this code of conduct, see the FAQ or its translations.

CHAPTER

NINE

SECURITY POLICY

9.1 Supported Versions

Version	Status
3.1.x	Stable version
3.0.x	Security fixes only
< 3.0	not supported

9.2 Reporting a Vulnerability

Please disclose security vulnerabilities privately at miurahr@linux.com

CHAPTER

TEN

AUTHORS

Aqtinstall is written and maintained by Hiroshi Miura <miurahr@linux.com>

Original qli-installer is written by Linus Jahn

Significant contributions for improvements of version 2.0 and 2.1 by David Dalcino David also leads many developments and reviews effort after 2.0.

All contributors, listed alphabetically, are:

- Alberto Mardegan(ignore_hash option)
- Andrei Yankovich (tools ifw installation)
- Aurélien Gâteau (patching to qmake)
- Benjamin O (Github Actions and more)
- Christian Hoffmann (Update mirror list)
- David Dalcino (Many improvements on CI automations, commands, tests, documents and so on)
- Doronin Stanislav (document)
- Fabrice Le Bars (32bit binary)
- Felix Barz (Android, Explicit extra module installation)
- Gamso (improve parsing of update.xml)
- Julien Marrec (mypy, type hints)
- Kyle Altendorf (7z binary path search)
- @lebarsfa (ignore_hash/hash_algorithm options)
- @lightmare (Documents)
- Martin Delille (Documents)
- Mike Tzou (Update fallback url)
- mite-user (folder index handling of download web sites)
- Mizux Seihax (Qt versions)
- Mozi (CI/workflow improvement, log format)
- Nelson Chen (CI tests)
- @nikitalita (Binary distribution)
- @pylipp (Documents)
- @Steveice10 (MacOS binary build)

- Sztergbaum Roman (Version database)
- Thomas Grainger (CLI entry point)
- @tsteven4 (fix patching to qmake, pkgconfig and libtool)
- Vadim Peretokin (Version database)
- Vladyslav Hnatiuk (Version database)
- @ypnos (Documents)

and many other participants and contributors. If you find a missing name to record, please feel free to tell me.

CHAPTER

ELEVEN

CHANGELOG

All notable changes to this project will be documented in this file.

11.1 Unreleased

11.2 v3.1.13 (13, Apr. 2024)

11.2.1 Added

- Add support for arm64 architecture on linux desktop (#766)

11.2.2 Changed

- Add Qt 6.6.3 as known version (#773)

11.2.3 Document

- Add example command line that show combinations of sub-commands (#759)

11.3 v3.1.12 (2, Mar. 2024)

11.3.1 Fixed

- Fix generating combination issue with Linux Qt 6.7 (#756,#757)

11.3.2 Added

- Add docs clarifying list-doc and install-doc (#754)

11.3.3 Changed

- Add Qt 6.7(#758)
- Update mingw variations (#758)
- Update IFW version to 47 (#763)
- Update Flake8@7.0.0

11.4 v3.1.11 (28, Nov. 2023)

11.4.1 Fixed

- Patch `*.prl` and `*.pc` for mingw (#640, #739)

11.4.2 Changed

- Add Qt 6.6.1 as known version (#740)
- chore: Improved CI to catch the problem with incorrect PRL files (#738)
- **chore: Update CI execution trigger/schedule (#735)**
  - **Full tests weekly on master**
    - mac, windows and linux
    - Qt 5.12.12, 5.15.14, 6.5.3
    - Python 3.9, 3.10, 3.11 and 3.12
    - check sample app built
  - **Change trigger for GitHub actions**
    - mac, windows and linux
    - Qt 4.9.9, 6.1.0
    - Python 3.9 and 3.12
    - check qmake works

11.5 v3.1.10 (14, Nov. 2023)

11.5.1 Fixed

- list_* commands ignore base url setting (#731,#732)

11.5.2 Changed

- chore: support build on git export (#730)

11.6 v3.1.9 (6, Nov. 2023)

11.6.1 Security

- CVE-2023-32681: Bump requests@2.31.0 (#724)

11.6.2 Changed

- Remove a specific mirror from fallback (#688)
- add `debug` extras for test and check (#725)
- Bump pytest-remotedata@0.4.1
- Bump flake8,flake8-isort@6.0.0 (#726)
- docs: change interpreted text to inline literals (#728)

11.6.3 Added

- macOS binary build (#722)
- `ignore_hash` and `hash_algorithm` options (#684)

11.7 v3.1.8 (1, Nov. 2023)

11.7.1 Changed

- Add 6.5.3 and openssl as known versions (#718)
- Docs: remove deprecated configuration description (#714)
- Test: test on python 3.8, 3.9 and 3.11 (#715)
- Docs: Update documentation for `--autodesktop` flag (#713)
- Use 'tar' filter when extracting tarfiles (#707)
- Log a warning when aqtinstall falls back to an external 7z extraction tool (#705)
- Bump py7zr@0.20.6(#702)

11.7.2 Fixed

- Fix failed CI (#716)
- Fix installation of win64_msvc2019_arm64 arch (#711)
- Fix `test_install` that fails on Python<3.11.4 (#708)
- Fix failing documentation builds (#706)
- Fix: exception when target path is relative (#702)

11.8 v3.1.7 (1, Aug. 2023)

11.8.1 Added

Add support for standalone sdktool installation(#677)

11.8.2 Fixed

- Fixed command to check tools_mingw90 (#680)
- Fixed help text for list-tool

11.8.3 Changed

- Add Qt 6.6.0, 6.5.2 and 6.5.1 as known version(#685,#698)
- Default blacklist setting(#689)
- Add test for sdktool(#678)

11.9 v3.1.6 (4, May, 2023)

11.9.1 Added

- Add opensslv3 as known module (#674)
- Add code signature for standalone binary

11.10 v3.1.5 (30, Mar. 2023)

11.10.1 Fixed

- Fix failure to install Qt 6.4.3 source and docs on Windows(#665)
- Fix failed .tar.gz extraction in `install-src` and `install-doc` (#663)

11.11 v3.1.4 (25, Mar. 2023)

11.11.1 Changed

- Add Qt 6.4.3 as known version(#661)
- Catch OSError(errno.ENOSPC) and PermissionError (#657)
- Update security policy

11.12 v3.1.3 (2, Mar. 2023)

11.12.1 Changed

- make the message about "unknown" Qt versions and modules more friendly and easy to understand (#646,#654)

11.13 v3.1.2 (17, Feb. 2023)

11.13.1 Fixed

- CI: Pin checkout at v3 in all workflows(#649)
- Fix list-qt and install-qt handling of WASM for Qt 6.5.0 (#648)

11.13.2 Changed

- Update combinations.xml (#650)
- Update documentation for `--autodesktop` flag (#638)

11.14 v3.1.1 (10, Feb. 2023)

11.14.1 Fixed

- CI: Pin EMSDK version (#641)
- Test: update tox.ini config (#634)
- Fix errors in install-* caused by duplicate modules (#633)

11.15 v3.1.0 (5, Dec. 2022)

11.15.1 Fixed

- Support Qt 6.4.1 Android installation (#621,#626,#627)
- Fix URL of Nelson's blog on README

11.15.2 Changed

- Update pyproject.toml and drop setup.cfg
- Standalone binary build with PyInstaller directly(#598)
- **Bump dependencies versions**
  - py7zr>=0.20.2
  - flake8<6
  - flake8-isort>=4.2.0
- metadata: change link to changelog
- docs: move CHANGELOG.rst into docs/
- Refactoring internals and now check types with mypy

11.15.3 Deprecated

- Drop support for python 3.6

11.16 v3.0.2 (26, Oct. 2022)

- Fix installation of Qt6/WASM arch on windows (#583,#584)
- Docs: allow localization (#588)
- Docs: Add Japanese translation (#595)

11.17 v3.0.1 (30, Sep. 2022)

- Actions: Fix standalone executable upload (#581)
- Actions: Bump versions (#579) - pypa/gh-action-pypi-publish@v1 - actions/setup-python@v4

11.18 v3.0.0 (29, Sep. 2022)

11.18.1 Added

- Automatically install desktop qt when required for android/ios qt installations(#540)

11.18.2 Fixed

- Tolerate empty DownloadArchive tags while parsing XML(#563)
- Fix standalone executable build for windows (#565,#567)

11.18.3 Changed

- Update Security policy
- Update combinations.json(#566)
- CI: now test on MacOS 12(#541)

11.19 v2.2.3 (17, Aug. 2022)

11.19.1 Fixed

- Building standalone executable: aqt.exe (#556,#557)

11.19.2 Added

- Docs: add explanation of `list-qt --long-modules` (#555)

11.20 v2.2.2 (11, Aug. 2022)

11.20.1 Added

- Add `aqt list-qt --long-modules` (#543,#547)

11.20.2 Fixed

- Fix kwargs passed up AqtException inheritance tree (#550)

11.20.3 v2.2.1 (9, Aug. 2022)

11.20.4 Changed

- `install-qt` command respect `--base` argument option when retrieve metadata XML files by making `MetadataFactory` respect `baseurl` set. (#545)

11.21 v2.2.0 (2, Aug. 2022)

11.21.1 Added

- Add code of conduct (#535)

11.21.2 Changed

- test: prevent use of flake8@5.0 (#544)
- Improve tox and pytest config(#544)
- Properly retrieve folder names from html pages of all mirrors(#520)
- Log: left align the level name (#539)
- Update combinations (#537)
- Introduce Updates.xml data class and parser (#533)
- archives: do not keep update.xml text in field (#534)
- docs: Bump sphinx@5.0 (#524)

11.21.3 Fixed

- Update readthedocs config (#535)
- Fix readme description of list-qt (#527)

11.21.4 Deprecated

- Deprecate setup.py file (#531)

11.22 v2.1.0 (14, Apr. 2022)

11.22.1 Changed

- Change security policy(#506): Supported 2.0.x Unsupported 1.2.x and before
- Bump py7zr@0.18.3(#509)
- pyproject.toml configuration * project section(#507) * setuptools_scm settings(#508)
- Use SHA256 hash from trusted mirror for integrity check (#493)

- Update combinations.xml * QtDesignStudio generation2 (#486) * IFW version (from 42 to 43) change (#495) * Support Qt 6.2.4 (#502)
- Update fallback mirror list (#485)

11.22.2 Fixed

- Fix patching of Qt6.2.2-ios(#510, #503)
- Test: Conditionally install dependencies on Ubuntu (#494)

11.22.3 Added

- doc: warn about unrelated aqt package (#490)
- doc: add explanation of –config flag in CLI docs (#491)
- doc: note about MSYS2/Mingw64 environment

11.22.4 Security

- Use secrets for secure random numbers(#498)
- Use defusedxml to parse Updates.xml file to avoid attack(#498)
- Improve get_hash function(#504)
- Check Update.xml file with SHA256 hash (#493)

CHAPTER

TWELVE

CHANGES UNTIL V2.0.6

12.1 v2.0.6 (7, Feb. 2022)

12.1.1 Fixed

- Fix archives flag(#459)
- Accept the case Update.xml in Server has delimiter without space(#479)
- Fix getUrl function to use property http session and retry(#473)

12.1.2 Added

- 32bit release binary(#471)

12.1.3 Changed

- Update combinations.xml * Qt 6.2.2, 6.2.3, 6.3.0(#481,#484)

12.2 v2.0.5 (11, Dec. 2021)

12.2.1 Changed

- Reduce memory consumption: garbage collection on install subprocess(#464)
- Cache PowerShell modules on Azure Pipeline(#465)

12.3 v2.0.4 (5, Dec. 2021)

12.4 Fixed

- Allow duplicated install on the directory previously installed(#438,#462)
- Memory error on 32bit python on Windows(#436,#462)

12.5 Changed

- Change list-src, list-doc and list-example command(#453)

12.6 v2.0.3 (25, Nov. 2021)

12.6.1 Added

- Improve –keep and new –archive-dest options(#458)

12.6.2 Fixed

- Fix cross-platform installation failure (#450)
- CI: update OSes, Windows-2019, macOS-10.15(#444,#456)
- CI: fix failure of uploading coveralls(#446)
- CI: test for QtIFW(#451)

12.6.3 Changed

- combinations matrix json(#452)

12.7 v2.0.2 (1, Nov. 2021)

12.7.1 Added

- Support Qt 6.2.1 (#441)

12.7.2 Fixed

- Degraded install-tool (#442,#443)

12.7.3 Changed

- Add suggestion to use `--external` for MemoryError (#439)

12.8 v2.0.1 (29, Oct. 2021)

12.8.1 Added

- Allow retries on checksum error(#420)
- Run on Python 3.10(#424)
- Add more mirrors for fallback(#432)
- Add fallback URL message(#434)

12.8.2 Fixed

- `--noarchives` inconsistency(#429)
- Allow multiprocessing error propagation(#419)
- Legacy command behavior, reproduce also old bugs (#414)
- Fix crash on `crash install-qt <host> <tgt> <spec>` with no specified arch(#435)

12.8.3 Changed

- Print working directory and version in error message(#418)

12.8.4 Security

- Use HTTPS for mirror site(#430)

12.9 v2.0.0 (29, Sep. 2021)

12.9.1 Added

- Add error messages when user inputs an invalid semantic version(#291)
- Security Policy document(#341)
- CodeQL static code analysis(#341)
- CI: generate combination json in actions (#318,#343)
- Test: add and improve unit tests(#327,#359)
- Docs: getting started section(#351)
- Docs: recommend python3 for old systems(#349)
- Automatically update combinations.json (#343,#344,#345,#386,#390,#395)
- CI: test with Qt6.2 with modules(#346)
- README: link documentation for stable(#329)
- Support WASM on Qt 6.2.0(#384)
- Add Binary distribution for Windows(#393,#397)

- Add list-qt –archives feature(#400)
- Require architecture when listing modules(#401)

12.9.2 Changed

- list subcommand now support tool information(#235)
- list subcommand can show versions, architectures and modules.(#235)
- C: bundle jom.zip in source(#295)
- Add max_retries configuration for connection(#296)
- Change settings.ini to introduce [requests] section(#297)
- Change log format for logging file.
- Extension validation for tool subcommand(#314)
- list subcommand has –tool-long option(#304, #319)
- tool subcommand now install without version spec(#299)
- README example command is now easy to copy-and-paste(#322)
- list subcommand update(#331)
- Improve handle of Ctrl-C keyboard interruption(#337)
- Update combinations.json(#344,#386)
- Turn warnings into errors when building docs(#360)
- Update documentations(#358,#357)
- Test: consolidate lint configuration to pyproject.toml(#356)
- Test: black configuration to max_line_length=125 (#356)
- New subcommand syntax (#354,#355)
- Failed on missing modules(#374)
- Failed on missing tools(#375)
- Remove ‘addons’ prefix for some modules for Qt6+ (#368)
- Fix inappropriate warnings(#370)
- Update README to fix version 2 (#377)
- list-qt: Specify version by SimpleSpec(#392)
- Add helpful error messages when modules/tools/Qt version does not exist(#402)

12.9.3 Fixed

- Fix helper.getUrl() to handle several response statuses(#292)
- Fix Qt 6.2.0 target path for macOS.(#289)
- Fix WinRT installation patching(#311)
- Fix Qt 5.9.0 installation (#312)
- Link documentations for stable/latest on README
- Check python version when starting command (#352)
- README: remove '$' from example command line(#321)
- README: fix command line example lexer(#322)
- CI: fix release script launch conditions(#298)
- Handle special case for Qt 5.9.0(#364)
- Running python2 -m aqt does not trigger Python version check (#372,#373)
- docs(cli): correct the parameter of "list-tool" in an example(#399)
- Doc: Fix broken mirror link in cli.rst (#403)
- CI: fix release action fails with no files found(#405)

12.10 v1.2.5 (14, Aug. 2021)

12.10.1 Fixed

- Handle Qt 5.9 installation special case(#363,#365)

12.11 v1.2.4 (17, Jul. 2021)

12.11.1 Fixed

- Fix crash when installing Qt6.1.2 on mac(#288,#320)

12.12 v1.2.3 (14, Jul. 2021)

12.12.1 Changed

- helper: set max_retries (#296)

12.12.2 Fixed

- Patching for winrt packages(#311)
- CI: Fix release note script
- CI: bundle jom.zip for test

12.13 v1.2.2 (1, Jul. 2021)

12.13.1 Added

- Create qtenv2.bat file on windows(#279)

12.13.2 Fixed

- Fix list subcommand to retrieve information from web(#280)
- Fix crash when installing Qt6.2.0 on mac(#288,#289)

12.14 v1.2.1 (22, Jun. 2021)

12.14.1 Fixed

- Fix crash when tool subcommand used.(#275,#276)

12.15 v1.2.0 (21, Jun. 2021)

12.15.1 Added

- Add -c/–config option to specify custom settings.ini(#246)
- Document for settings.ini configuration parameters(#246)
- Patching libtool file(.la) on mac(#267)
- CI: Add more blacklist mirrors
- Add –kde option for src subcommand(#274)

12.15.2 Changed

- Use spawn multiprocessing on Linux platform.(#273)
- Check MD5 checksum when download(#238)
- Config settings.ini parser and URL list format(#246)
- Refactoring network connection code, consolidated to helper.py(#244)
- Refactoring exceptions, introduce exceptions.py(#244)

- Update known Qt versions combinations.(#243)
- CI: changes azure pipelines test scripts(#250)

12.15.3 Fixed

- Fix logging during subprocess installation on macOS, and Windows(#273)
- Fix patching qmake(#259)
- Prettify help message format(#237)
- Update patching pkgconfig/lib on mac(#267)
- CI: fix check workflow(#248)
- CI: fix error on Azure/Windows(connection error)(#246)
- Fix typo in README(#326)

12.16 v1.1.6 (2, May. 2021)

12.16.1 Fixed

- doc subcommand failed in argument parse(#234)

12.17 v1.1.5 (8, Apr. 2021)

12.17.1 Added

- README: describe advanced installation method.

12.17.2 Changed

- Change tox.ini: docs test output folder
- Remove changelog from pypi page

12.17.3 Fixed

- Drop dependency for wheel

12.18 v1.1.4 (2, Apr. 2021)

12.18.1 Changed

- Code reformatting by black and check by black.
- Check linting by github actions.

12.18.2 Fixed

- Fix document error on README(#228, #226).

12.19 v1.1.3 (26, Feb. 2021)

12.19.1 Fixed

- Key error on 3.6.13, 3.7.10, 3.8.8, and 3.9.2(#221)

12.20 v1.1.2 (20, Feb. 2021)

12.20.1 Fixed

- Fix leaked multiprocessing resource(#220)
- Catch both read timeout and connection timeout.

12.21 v1.1.1 (13, Feb. 2021)

12.21.1 Fixed

- Catch timeout error and fallback to mirror (#215,#217)

12.22 v1.1.0 (12, Feb. 2021)

Added ——.. _v2.0.1: https://github.com/miurahr/aqtinstall/compare/v2.0.0...v2.0.1

- Patching android installation for Qt6 - patch target_qt.conf

12.22.1 Changed

- CI test with Qt6
- Docs: update avaiable conbinations

12.22.2 Fixed

- Skip QtCore patching for 5.14.0 and later(Fix regression)(#211)

12.23 v1.0.0 (4, Feb. 2021)

12.23.1 Added

- Add –noarchives option to allow user to add modules to existed installation(#174,#204)
- No patching when it does not install qtbase package by –noarchives and –archives option.(#204)
- Azure: test with jom build on windows.
- Patch pkgconfig configurations(#199)
- Patch libQt5Core and libQt6Core for linux(#201)

12.23.2 Changed

- Update document to show available Qt versions
- Update README to add more references.
- Suppress debug log and exist silently when specified package not found.

12.23.3 Fixed

- Catch exception on qmake -query execution(#201)
- Fix Qt6/Android installation handling.(#193, #200)

12.24 v0.11.1 (21, Jan. 2021)

12.24.1 Added

- Add –timeout option to specify connection timeout (default 5.0 sec) (#197)

12.25 v0.11.0 (21, Jan. 2021)

12.25.1 Added

- Automatically fallback to mirror site when main https://download.qt.io down.(#194, #196)

12.26 v0.10.1 (11, Dec. 2020)

12.26.1 Added

- Add LTS versions as known one.(#188)

12.26.2 Changed

- Tool: Version comparison by startswith. When specified 4.0 but download server hold 4.0.1, it catch 4.0.1.(related #187)
- README: explicitly show python version requirements.

12.27 v0.10.0 (25, Nov. 2020)

12.27.1 Added

- Add v5.12.2, v6.0.0 as known versions.(#176, #177)
- Support –archives option on src installation.

12.27.2 Changed

- Use multiprocessing.Pool instead of concurrent.futures(#178)
- Refactoring whole modules. (#179)
- Split old changelogs to CHNAGELOG_prerelease.rst
- Drop an upper limitation (<0.11) for py7zr.(#183)

12.27.3 Fixed

- When we used "-m all" to download doc or examples, Qt sources are also downloaded(@Gamso)(#182)

12.28 v0.9.8 (4, Nov. 2020)

12.28.1 Added

- Added new combinations for tools_ifw

12.29 v0.9.7 (4, Oct. 2020)

12.29.1 Added

- Support Qt 5.15.1
- Add list command. (#161)

12.29.2 Fixed

- When we start an installation, all packages are downloaded whatever the specified platform.(#159)

12.30 v0.9.6 (7, Sep. 2020)

12.30.1 Changed

- setup: set minimum required python version as >=3.6.

12.31 v0.9.5 (18, Aug. 2020)

12.31.1 Fixed

- Fix error when install tools_openssl_src (#153)

12.32 v0.9.4 (2, Aug. 2020)

12.32.1 Fixed

- Fixed CRC32 error when installing Qt5.7.(#149)

12.33 v0.9.3 (1, Aug. 2020)

12.33.1 Fixed

- **Fixed failure when installing Qt5.7 which archives use 7zip(LZMA1+BCJ), that is supported** by py7zr v0.9 and later.(#149,#150)

12.34 v0.9.2 (19, June. 2020)

12.34.1 Added

- Add Qt6 as a known version. (#144)

12.34.2 Fixed

- Support package directory 'qt6_*' as well as 'qt5_*' (#145)

12.35 v0.9.1 (13, June. 2020)

12.35.1 Changed

- Do not raise exception when specified combination is not found on downloaded meta data.
- Update document for developers.

12.36 v0.9.0 (31, May. 2020)

12.36.1 Added

- New subcommand doc/src/example to install each components.(#137, 138)
- Doc: Add CLI example for tools, doc, examples and src.

12.36.2 Changed

- Refactoring to reduce code duplication in archives.py
- Explicitly call QtInstall.finalize() only when Qt library installation.

12.36.3 Fixed

- Show help when launched without any argument (#136)

12.37 v0.9.0b3 (21, May. 2020)

12.37.1 Changed

- Patch qmake when finishing installation.(#100) qmake has a hard-coded prefix path, and aqt modify binary in finish phase. it is not necessary for Qt 5.14.2, 5.15.0 and later. This behavior try to be as same as a Qt installer framework doing.
- Patch Framework.QtCore when finishing installation.(#100) As same as qmake, framework also has a hard-coded prefix path. (Suggestions from @agateau)

12.38 v0.9.0b2 (21, May. 2020)

12.38.1 Added

- CLI: '–archives' option: it takes multiple module names such as qtbase, qtsvg etc. This is an advanced option to specify subset of target installation. There is no guarantee it works. It is not recommended if you are unknown what is doing.

12.39 v0.9.0b1 (10, May. 2020)

12.39.1 Added

- Support installation of Qt version for msvc2019
- Add knowlege of components combination on 5.14 and 5.15

12.39.2 Changed

- Show detailed diagnose message when error happend.
- CI test with Qt 5.14.2 and 5.15.0
- CI test with installed mingw tools compiler.
- Depends on py7zr v0.7.0b2 and later.

12.39.3 Fixed

- Tools: Fix mingw installation failure.
- Fix –outputdir behavior about path separator on windows

12.40 v0.8 (26, Mar. 2020)

12.40.1 Fixed

- docs: fix broken link for qli-installer

12.41 v0.8b1 (12, Mar. 2020)

12.41.1 Added

- Support specifing config with environment variable AQT_CONFIG

12.41.2 Fixed

- Fix to use concurrency settings

12.42 v0.8a4 (6, Mar., 2020)

12.42.1 Fixed

- Import-metadata package is required in python version < 3.8 not 3.7.
- Refactoring redirect helper function to improve connection error checks and error message.(#109)

12.43 v0.8a3 (5, Mar., 2020)

12.43.1 Changed

- Improve error messages when command argument is wrong.

12.43.2 Fixed

- Work around for https://download.qt.io/ returns wrong metalink xml data.(#105, #106)

12.44 v0.8a1 (28, Feb., 2020)

12.44.1 Changed

- Allow path search for 7z (#96)
- Simplify multithreading using concurrent.futures.ThreadPoolExecutor().

12.44.2 Fixed

- Detect exception on each download and extraction threads.
- Race condition error happend on py7zr. require py7zr>=0.5.3.(#97)

12.45 v0.7.4 (15, Feb., 2020)

12.45.1 Changed

- requirement of py7zr version become >0.6b2 which fixed a multiprocessing problem.

12.46 v0.7.3 (14, Feb., 2020)

12.46.1 Added

- Github Actions workflows for publishing.

12.46.2 Changed

- Remove run script from source. Now it is automatically generated when build.(#85)
- Update requirement py7zr >=0.5

12.46.3 Fixed

- README: fix reStructured text syntax.

12.47 v0.7.2 (11, Feb., 2020)

12.47.1 Changed

- **Replace 'multiprocessing.dummy' with 'concurrent.futures'.**
  - download with multi-threading(I/O bound)
  - extract with multi-processing(CPU bound)

12.47.2 Fixed

- '-E | –external' option handling which cause path is not str error.

12.48 v0.7.1 (13, Jan., 2020)

12.48.1 Fixed

- Fix installation of extra modules for Qt5.9.

12.49 v0.7 (13, Jan., 2020)

12.49.1 Changed

- Move project metadata to setup.cfg from setup.py.

12.50 v0.7b1 (10, Jan., 2020)

12.50.1 Changed

- Bamp up dependency py7zr >=v0.5b5.
- Use py7zr in default to extract packages.
- Drop –internal command line option.

12.51 v0.7a2 (7, Jan., 2020)

12.51.1 Added

- Add special module name 'all' for extra module option.

12.51.2 Fixed

- CI conditions, update target version.

12.52 v0.7a1 (29, Nov., 2019)

12.52.1 Added

- Introduce helper module.
- Introduce 'settings.ini' file which has a configuration for aqt module.

12.52.2 Changed

- Version numbering with setuptools_scm.
- Now don't install extra modules when installing 'wasm_32' arch. You should explicitly specify it with '-m' option.

12.52.3 Fixed

- Error when mirror site is not http, but https and ftp.

12.53 v0.6b1 (23, Nov., 2019)

12.53.1 Changed

- Just warn when argument combination check is failed.
- CI: Compress sample project for build test with 7zip.
- CI: Place sample script in ci directory.

12.54 v0.6a2 (19, Nov., 2019)

12.54.1 Added

- Test: Unit test against command line.
- Android target variants.

12.54.2 Changed

- Use logging configuration with logging.ini

12.54.3 Fixed

- qconfig.pri: fix QT_LICHECK line.

12.54.4 Removed

- Logging configuration file logging.yml
- Drop dependency for pyyaml.

12.55 v0.6a1 (17, Nov., 2019)

12.55.1 Added

- More build test with sample project which uses an extra module.(#56)
- Add support for installation of WebAssembly component by specifying 'wasm_32' as an arch argument.(#53, #55)

12.55.2 Changed

- Optional modules are installed explicitly. Users need to specify extra modules with -m option.(#52, #56)

12.55.3 Fixed

- Dependency for py7zr only for python > 3.5. Now it works with python2.7.

12.56 v0.5 (10, Nov., 2019)

12.56.1 Changed

- Introduce combination DB in json form. User and developer now easily add new component for installation checking.

12.56.2 Fixed

- requires py7zr >= 0.4.1 because v0.4 can fails to extract file.

12.57 v0.5b2 (8, Oct., 2019)

12.57.1 Changed

- Change install path from <target>/Qt/Qt<version>/<version> to <target>/<version> (#48). - Also update CI test to specify –outputdir <target> that is $(BinariesDirectory)/Qt

12.58 v0.5b1 (8, Oct., 2019)

12.58.1 Added

- Add feature to support installation of Qt Tools
- Add CI test for tool installation

12.58.2 Changed

- CI test target - add 5.14.0 - remove 5.11.3 - change patch_levels to up-to-date

12.59 v0.4.3 (25, Sep, 2019)

12.59.1 Fixed

- Allow multiple redirection to mirror site.(#41)

12.60 v0.4.2 (28, Jul, 2019)

12.60.1 Changed

- README: update badge layout.
- CI: Improve azure-pipelines configurations by Nelson (#20)
- Check parameter combination allowance and add winrt variant.
- Support installation of mingw runtime package.
- Add '–internal' option to use py7zr instead of external 7zip command for extracting package archives.(WIP)

12.61 v0.4.1 (01, Jun, 2019)

12.61.1 Added

- Option -b | –base to specify mirror site.(#24)

12.61.2 Changed

- CI: add script to generate auzre-pipelines.yml (#27, #28, #29)
- CI: use powershell script for linux, mac and windows. (#26)

12.61.3 Fixed

- Avoid blacklisted mirror site that cause CI fails.(#25)

12.62 v0.4.0 (29, May, 2019)

12.62.1 Added

- cli: output directory option.
- sphinx document.
- test packaging on CI.
- Handler for metalink information and intelligent mirror selection.

12.62.2 Changed

- Change project directory structure.
- cli command name changed from 'aqtinst' to 'aqt' and now you can run 'aqt install'
- Introduce Cli class
- Massive regression test on azure pipelines(#20)
- blacklist against http://mirrors.tuna.tsinghua.edu.cn and http://mirrors.geekpie.club/ from mirror site.
- Run 7zip command with '-o{directory}' option.

12.62.3 Fixed

- Fix File Not Found Error when making qt.conf against win64_mingw73 and win32_mingw73

12.63 v0.3.1 (15, March, 2019)

12.63.1 Added

- Qmake build test code in CI environment.(#14)

12.63.2 Fixed

- Connect to Qt download server through proxy with authentication.(#17)

12.63.3 Changed

- Change QtInstaller.install() function signature not to take any parameter.
- Replace standard urllib to requests library.(#18)
- Use 7zr external command instead of 7z in Linux and mac OSX envitonment.

12.63.4 Removed

- requirements.txt file.

12.64 v0.3.0 (8, March, 2019)

12.64.1 Added

- Allow execute both 'aqtinst' and 'python -m aqt' form.

12.64.2 Changed

- Project URL is changed.
- Generate universal wheel support both python2.7 and python 3.x.

12.64.3 Fixed

- Update README wordings.
- Remove dependency for python3 with 'aqtinst' command utility.
- Fix command name in help message.

12.65 v0.2.0 (7, March, 2019)

12.65.1 Added

- Released on pypi.org

12.65.2 Changed

- Install not only basic packages also optional packages.
- Rename project/command to aqt - Another QT installer

12.65.3 Fixed

- Update mkspecs/qconfig.pri to indicate QT_EDITION is OpenSource
- Support Python2

12.66 v0.1.0 (5, March, 2019)

12.66.1 Changed

- Support multiprocess concurrent download and installation.

12.67 v0.0.2 (4, March, 2019)

12.68 Added

- CI test on Azure-pipelines

12.69 Changed

- Refactoring code
- Install QtSDK into (cwd)/Qt<version>/<version>/gcc_64/
- Drop dependency for requests library
- Use standard argparse for command line argument.

12.70 Fixed

- Support windows.
- looking for 7zip in standard directory.

12.71 v0.0.1 (2, March, 2019)

- Fork from qli-installer

CHAPTER

THIRTEEN

INDICES AND TABLES

- genindex
- modindex
- search

INDEX

Symbols

-E
    list-tool command line option, 24
-O
    list-tool command line option, 24
--arch
    list-qt command line option, 21
--archive-dest
    list-tool command line option, 24
--archives
    list-qt command line option, 21
    list-tool command line option, 25
--autodesktop
    install-qt command line option, 26
--base
    list-tool command line option, 24
--config
    list-tool command line option, 24
--external
    list-tool command line option, 24
--help
    list-qt command line option, 19
    list-tool command line option, 23, 24
--internal
    list-tool command line option, 24
--kde
    install-src command line option, 27
--keep
    list-tool command line option, 24
--latest-version
    list-qt command line option, 21
--long
    list-tool command line option, 23
--long-modules
    list-qt command line option, 20
--modules
    list-doc command line option, 22
    list-example command line option, 23
    list-qt command line option, 20
    list-tool command line option, 24
--noarchives
    install-qt command line option, 26
--outputdir
    list-tool command line option, 24
--spec
    list-qt command line option, 20
--timeout
    list-tool command line option, 24
-b
    list-tool command line option, 24
-c
    list-tool command line option, 24
-h
    list-qt command line option, 19
    list-tool command line option, 23, 24
-k
    list-tool command line option, 24
-l
    list-tool command line option, 23
-m
    list-tool command line option, 24

I

install-qt command line option
    --autodesktop, 26
    --noarchives, 26
install-src command line option
    --kde, 27
install-tool command line option
    tool, 29

L

list-doc command line option
    --modules, 22
list-example command line option
    --modules, 23
list-qt command line option
    --arch, 21
    --archives, 21
    --help, 19
    --latest-version, 21
    --long-modules, 20
    --modules, 20
    --spec, 20

-h, 19
list-tool command line option
-E, 24
-O, 24
--archive-dest, 24
--archives, 25
--base, 24
--config, 24
--external, 24
--help, 23, 24
--internal, 24
--keep, 24
--long, 23
--modules, 24
--outputdir, 24
--timeout, 24
-b, 24
-c, 24
-h, 23, 24
-k, 24
-l, 23
-m, 24

T

tool
install-tool command line option, 29