aqtinstall Documentation

Release 2.1.0

Hiroshi Miura

Apr 13, 2022

CONTENTS

Contents:

CHAPTER

ONE

INSTALLATION

1.1 Requirements

- Minimum Python version: 3.6
- Recommended Python version: 3.7.5 or later
- Dependent libraries: requests, py7zr, semantic_version, patch, texttable, bs4

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, or linux)
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.

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
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
```

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
```

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 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*. This exercise is left to the reader.

3.5 Installing Qt for Android

Let's install Qt for Android. Installing Qt 5 will be similar to installing Qt for Desktop on Windows, but there will be differences when we get to Qt 6.

```
$ aqt list-qt windows android                          # Print Qt versions available
5.9.0 5.9.1 ...
...
6.2.0

$ aqt list-qt windows android --arch 5.15.2            # Print architectures available
android

$ aqt list-qt windows android --modules 5.15.2 android  # Print modules available
qtcharts qtdatavis3d qtlottie qtnetworkauth qtpurchasing qtquick3d qtquicktimeline␣
↪qtscript

$ aqt install-qt windows android 5.15.2 android -m qtcharts qtnetworkauth   # Install
```

Let's see what happens when we try to list architectures and modules for Qt 6:

```
$ aqt list-qt windows android --arch 6.2.0                    # Print architectures␣
↪available
Command line input error: Qt 6 for Android requires one of the following extensions:
('x86_64', 'x86', 'armv7', 'arm64_v8a').
Please add your extension using the `--extension` flag.

$ aqt list-qt windows android --modules 6.2.0 android_armv7     # Print modules available
Command line input error: Qt 6 for Android requires one of the following extensions:
('x86_64', 'x86', 'armv7', 'arm64_v8a').
Please add your extension using the `--extension` flag.
```

The Qt 6 for Android repositories are a little different than the Qt 5 repositories, and the *aqt list-qt* tool doesn't know where to look for modules and architectures if you don't tell it what architecture you need. I know, it sounds a little backwards, but that's how the Qt repo was put together.

There are four architectures available, and the error message from *aqt list-qt* just told us what they are: *x86_64*, *x86*, *armv7*, and *arm64_v8a*.

We know we want to use *armv7* for the architecture, but we don't know exactly what value for 'architecture' we need to pass to *aqt install-qt* yet, so we will use *aqt list-qt* again:

```
$ aqt list-qt windows android --extension armv7 --arch 6.2.0
android_armv7
```

You should be thinking, "Well, that was silly. All it did was add *android_* to the beginning of the architecture I gave it. Why do I need to use `aqt list-qt --arch` for that?" The answer is, `aqt list-qt --arch` is checking to see what actually exists in the Qt repository. If it prints an error message, instead of the obvious *android_armv7*, we would know that Qt 6.2.0 for that architecture doesn't exist for some reason, and any attempt to install it with *aqt install-qt* will fail.

If we want to install Qt 6.2.0 for armv7, we use this command to print available modules:

```
$ aqt list-qt windows android --extension armv7 --modules 6.2.0 android_armv7
qt3d qt5compat qtcharts qtconnectivity qtdatavis3d qtimageformats qtlottie
qtmultimedia qtnetworkauth qtpositioning qtquick3d qtquicktimeline
qtremoteobjects qtscxml qtsensors qtserialbus qtserialport qtshadertools
qtvirtualkeyboard qtwebchannel qtwebsockets qtwebview
```

Finally, let's install Qt 6.2.0 for Android armv7 with the `qtcharts` and `qtnetworkauth` modules:

```
$ aqt install-qt linux android 6.2.0 android_armv7 -m qtcharts qtnetworkauth
```

3.6 Installing Qt for WASM

To find out how to install Qt for WASM, we need to tell *aqt list-qt* that we are using the *wasm* architecture. We can do that by using the `--extension wasm` flag.

```
$ aqt list-qt windows desktop --extension wasm
5.13.1 5.13.2
5.14.0 5.14.1 5.14.2
5.15.0 5.15.1 5.15.2
```

There are only a few versions of Qt that support WASM, and they are only available for desktop targets. If we tried this command with *android*, *winrt*, or *ios* targets, we would have seen an error message.

We can check the architecture and modules available as before:

```
$ aqt list-qt windows desktop --extension wasm --arch 5.15.2        # available␣
↪architectures
wasm_32

$ aqt list-qt windows desktop --extension wasm --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
```

3.7 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_ifw
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.

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

1. The host operating system (windows, mac, or linux)
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.

3.8 Installing a subset of Qt archives [Advanced]

3.8.1 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!

3.8.2 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.

3.8.3 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
```

3.8.4 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.

3.8.5 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
```

(continues on next page)

(continued from previous page)

```
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]
            [--extension <extension>]
            [--spec <specification>]
            [--modules    (<Qt version> | latest) <architecture> |
             --extensions (<Qt version> | latest) |
             --arch       (<Qt version> | latest) |
             --archives   (<Qt version> | latest) architecture [modules...]
             --latest-version]
           <host> [<target>]
```

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

host

linux, 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

`--extension` <Extension>

Extension of packages to list {wasm,src_doc_examples,preview,wasm_preview,x86_64,x86,armv7,arm64_v8a}

Use the `--extensions` flag to list all relevant options for a host/target. Incompatible with the `--extensions` flag, but may be combined with any other flag.

`--extensions` (<Qt version> | latest)

Qt version in the format of "5.X.Y", or the keyword `latest`. When set, this prints all valid arguments for the `--extension` flag for Qt 5.X.Y, or the latest version of Qt if `latest` is specified. Incompatible with the `--extension` flag.

`--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/extension/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.

`--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/extension, 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 `--extension` and/or `--spec` flags.

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, 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, 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, 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, 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>...]]
    [--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, 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.

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 for linux 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.

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

Install Qt documentation for the specified version and target.

host

linux, 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, 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, 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 5.12.0 for Linux with QtCharts and QtNetworkAuth:

```
pip install aqtinstall
sudo aqt install-qt --outputdir /opt linux desktop 5.12.0 -m qtcharts qtnetworkauth
```

Example: Installing the newest LTS version of Qt 5.12:

```
pip install aqtinstall
sudo aqt install-qt linux desktop 5.12 win64_mingw73
```

Example: Installing Android (armv7) Qt 5.10.2:

```
aqt install-qt linux android 5.10.2 android_armv7
```

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: 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 tools_vcredist
.\Qt\Tools\vcredist\vcredist_msvc2019_x64.exe /norestart /q
```

Example: Install MinGW on Windows

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

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

[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

[kde_patches]
```

(continues on next page)

(continued from previous page)

```
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.

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. When you are happy with the default sites, you can override with your 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

AQTINSTALL CHANGELOG

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

7.1 Unreleased

7.2 v2.1.0 (14, Apr. 2022)

7.2.1 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)

7.2.2 Fixed

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

7.2.3 Added

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

7.2.4 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)

7.3 v2.0.6 (7, Feb. 2022)

7.3.1 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)

7.3.2 Added

- 32bit release binary(#471)

7.3.3 Changed

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

7.4 v2.0.5 (11, Dec. 2021)

7.4.1 Changed

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

7.5 v2.0.4 (5, Dec. 2021)

7.6 Fixed

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

7.7 Changed

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

7.8 v2.0.3 (25, Nov. 2021)

7.8.1 Added

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

7.8.2 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)

7.8.3 Changed

- combinations matrix json(#452)

7.9 v2.0.2 (1, Nov. 2021)

7.9.1 Added

- Support Qt 6.2.1 (#441)

7.9.2 Fixed

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

7.9.3 Changed

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

7.10 v2.0.1 (29, Oct. 2021)

7.10.1 Added

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

7.10.2 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)

7.10.3 Changed

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

7.10.4 Security

- Use HTTPS for mirror site(#430)

7.11 v2.0.0 (29, Sep. 2021)

7.11.1 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)

7.11.2 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)

7.11.3 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)

7.12 v1.2.5 (14, Aug. 2021)

7.12.1 Fixed

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

7.13 v1.2.4 (17, Jul. 2021)

7.13.1 Fixed

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

7.14 v1.2.3 (14, Jul. 2021)

7.14.1 Changed

- helper: set max_retries (#296)

7.14.2 Fixed

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

7.15 v1.2.2 (1, Jul. 2021)

7.15.1 Added

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

7.15.2 Fixed

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

7.16 v1.2.1 (22, Jun. 2021)

7.16.1 Fixed

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

7.17 v1.2.0 (21, Jun. 2021)

7.17.1 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)

7.17.2 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)

7.17.3 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)

7.18 v1.1.6 (2, May. 2021)

7.18.1 Fixed

- doc subcommand failed in argument parse(#234)

7.19 v1.1.5 (8, Apr. 2021)

7.19.1 Added

- README: describe advanced installation method.

7.19.2 Changed

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

7.19.3 Fixed

- Drop dependency for wheel

7.20 v1.1.4 (2, Apr. 2021)

7.20.1 Changed

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

7.20.2 Fixed

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

7.21 v1.1.3 (26, Feb. 2021)

7.21.1 Fixed

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

7.22 v1.1.2 (20, Feb. 2021)

7.22.1 Fixed

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

7.23 v1.1.1 (13, Feb. 2021)

7.23.1 Fixed

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

7.24 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

7.24.1 Changed

- CI test with Qt6
- Docs: update avaiable conbinations

7.24.2 Fixed

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

7.25 v1.0.0 (4, Feb. 2021)

7.25.1 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)

7.25.2 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.

7.25.3 Fixed

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

CHAPTER

EIGHT

CONTRIBUTION GUIDE

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

8.1 Resources

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

8.2 Bug triage

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

8.3 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.

8.4 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

NINE

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

All contributors, listed alphabetically, are:

- Andrei Yankovich (tools ifw installation)
- Aurélien Gâteau (patching to qmake)
- Benjamin O (Github Actions and more)
- Christian Hoffmann (Update mirror list)
- David Dalcino (CI automations, List commands, improve tests, documents and fix patching to qmake)
- 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)
- lightmare (Documents)
- Mike Tzou (Update fallback url)
- Martin Delille (Documents)
- Mizux Seihax (Qt versions)
- Mozi (CI/workflow improvement)
- Nelson Chen (CI tests)
- @nikitalita (Binary distribution)
- @pylipp (Documents)
- 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

TEN

INDICES AND TABLES

- genindex
- modindex
- search

INDEX

Symbols

I

L

T