aqtinstall Documentation

リリース *3.1.11*

Hiroshi Miura

2023 年 11 月 28 日

目次

Contents:

第 1 章

インストール

1.1 要件

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

1.2 pip コマンドによるインストール

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

```
$ pip install aqtinstall
```

第 2 章

コマンドの変更

バージョン 2.0.0 から、サブコマンドが変更されました。これらのサブコマンドの以前のバージョンは、下位互換性のために保持されていますが、お勧めできません。

新しいサブコマンド	レガシーサブコマンド	Note
install-qt	install	バージョンはターゲットの後に移動しました
install-tool	tool	引数が変更されました 新しい構文はバージョンを使用しません
install-example	examples	バージョンはターゲットの後に移動しました 最後の引数に注意
install-src	src	バージョンはターゲットの後に移動しました 新しいコマンドは--kde オプションしか使えません
install-doc	doc	バージョンはターゲットの後に移動しました
	list	従来のリストコマンドは削除されました。
list-qt		
list-tool		

第 3 章

スタートアップガイド

aqt は、Qt、モジュール、Qt に関連するツール、ソース、ドキュメント、サンプルなどをインストールするためのツールで、https://download.qt.io/ から入手できます。aqt を実行する前に、何をインストールしたいのかを正確に aqt に指示する必要があります。このセクションでは、どのようなパッケージが aqt に用意されているのかを調べ、何をインストールしたいのかを aqt に指示できるようにするための手順を説明します。

すべての aqt サブコマンドには --help オプションがあります。なにか疑問が発生した場合は、このオプションを使用してください!

3.1 Qt をインストールする

``aqt``の一般的な使い方は以下のようになります:

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

pip と一緒に``aqt``をインストールしている場合は、コマンドスクリプト``aqt``で実行することができますが、場合によっては``python-m aqt``として実行する必要があるかもしれません。古いオペレーティングシステムでは、``python3-m aqt``のように Python バージョン 3 を指定する必要があるかもしれません。

Qt をインストールするには、aqt に次の 4 つのことを伝える必要があります。

1. ホストオペレーティングシステム (Windows、Mac、Linux)
2. ターゲット SDK(デスクトップ、android、ios、winrt)
3. インストールしたい Qt のバージョン
4. ターゲット・アーキテクチャ

Qt for IOS は Mac OS でのみ利用可能であり、Qt for WinRT は Windows でのみ利用可能であることに注意してください。

利用可能な Qt のバージョンを調べるには、 *aqt list-qt command* を使うことができます。このコマンドは Windows デスクトップで利用可能な Qt のすべてのバージョンを出力します:

```
$ 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
```

バージョン番号がソートされ、マイナーバージョン番号でグループ化され、1 つのスペース文字で区切られていることに注意してください。すべての *aqt list-qt* コマンドの出力は、 *aqt list-qt* の出力を消費するプログラムを作成しやすくするためのものです。

aqt list-qt コマンドは https://download.qt.io/ にある Qt ダウンロードリポジトリに直接問い合わせますので、このコマンドの結果は常に正確になります。Available Qt versions wiki ページは過去のある時点での更新ですので、最新であるかどうかは保障されません。

使用可能な Qt のバージョンがわかったので、バージョン 6.2.0 を選択します。

次にすべきことは、Qt 6.2.0 for Windows Desktop で利用可能なアーキテクチャを見つけることです。これを行うには、*aqt list-qt* に `--arch` フラグを付けて確認します:

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

これは Available Qt versions wiki ページにリストされているアーキテクチャの非常に小さなサブセットであることに注意してください。このリストにないアーキテクチャを使用する必要がある場合は、Available Qt versions wiki ページを使用して、どのバージョンが希望するアーキテクチャをサポートしているかの大まかなアイデアを得てから、 *aqt list-qt* を使用して、アーキテクチャが使用可能であることを確認します。

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
```

Qt 6.2 の次のバージョンが利用可能になったらすぐにインストールしたいとしましょう。これを行うには、明示的なバージョンではなく SimpleSpec を使用します。

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

3.2 7-Zip 解凍 外部 コマンド

デフォルトでは、aqt は Qt リポジトリに保存されている 7-zip アーカイブを、 aqt と一緒にインストールされている py7zr を使って抽出します。代わりに、 -E または --external フラグを使って 7-zip コマンドパスを指定することもできます。例えば、Windows デスクトップで次のコマンドを使って 7-zip を使うことができます。

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

Linux では、次のコマンドを使用して p7zip (Linux に移植された 7-zip) を指定できます。これはデフォルトでインストールされることが多い 7-zip です。

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

3.3 出力ディレクトリの変更

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.

別の場所にインストールしたい場合は、-O または --outputdir フラグを使う必要があります。このオプションは、aqt install- で始まる他のすべてのサブコマンドにも使えます。

純正の GUI インストーラが使うデフォルトのディレクトリである C:\Qt にインストールするには、以下のコマンドを使います:

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

3.4 モジュールのインストール

Qt 5.15.2 用のモジュールを Windows デスクトップにインストールする必要があるとします。まず、モジュールが何と呼ばれているかを調べる必要があります。そのためには *aqt list-qt* と --modules フラグを使用します。Qt の各バージョンには、ホスト OS/ターゲット SDK/アーキテクチャの組み合わせごとに異なるモジュールリストがあるので、その情報を *aqt list-qt* に提供する必要があります:

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

これらのモジュールをインストールする前に、これらのモジュールについてもっと知りたいとしましょう。そのためには `--long-modules` フラグを使うことができます:

```
$ 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
=========================================================================
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
```

利用可能なすべてのモジュールをインストールしたい場合は、`all` キーワードでインストールできます。

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

aqt list-qt コマンドは、スクリプト化できるように意図して作られていることを覚えていますか?Qt 5.15.2 で利用可能なすべてのモジュールをインストールする 1 つの方法は *aqt list-qt* の出力を *aqt install-qt* に以下のように送ることです:

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

このコマンドを実行するには Unix スタイルのシェルが必要ですし、Windows では少なくとも git-bash が必要です。この `xargs` に相当するコマンドは、読者の練習課題です。

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 Android 用 Qt のインストール

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

注釈: 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

$ aqt list-qt windows android --modules 6.2.4 android_armv7   # Print modules available
```

(次のページに続く)

(前のページからの続き)

```
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 WASM 用 Qt のインストール

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
```

以前と同じように Qt for WASM をインストールできます。

```
$ 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 ツールをインストールする

aqt list-tool コマンドを使って、Windows Desktop で利用できるツールを調べてみましょう。

```
$ 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
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
```

このコマンドの出力は、以下の *aqt install-tool* で使える値のリストになります。以下のように、`--l` または `--long` フラグを使用して詳細を調べてみましょう。

```
$ 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  ␣
↪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. ホストオペレーティングシステム (Windows、Mac、Linux)
2. ターゲット SDK(デスクトップ、android、ios、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
```

`aqt install-tool` は、各ツールに関連する ``installscript.qs`` を認識しないことに注意してください。これらのツールを標準の GUI インストーラでインストールする場合、インストーラは ``installscript.qs`` スクリプトを使ってシステムに追加の変更を加えることができます。これらの変更が必要な場合、`aqt` はこのスクリプトを実行することができないので、変更を加えるのはあなたの責任になります。

3.8 Qt アーカイブのサブセットをインストールする [上級編]

3.8.1 はじめに

お気づきかもしれませんが、デフォルトでは `aqt install-qt` は多くのアーカイブをインストールしますが、その必要はないかもしれませんし、一般的なインストールでは必要以上のディスク容量を消費します。モジュール `debug_info` をインストールした場合、1 GB 以上のデータがインストールされている可能性があります。この節では、Qt インストールのフットプリントを減らすための手助けをします。

注釈: 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 Qt の最小インストール

通常、`aqt install-qt` を実行すると、このプログラムはダウンロード、抽出、インストールしているアーカイブの長いリストを出力します。このリストには、`qtbase`、`qtmultimedia`、`qt3d`、そして約 25 項目が含まれています。これらのアーカイブのどれを実際にインストールするかは、`--archives` フラグを使って選択することができます。 `--archives` フラグは、Qt の基本インストールモジュールと ``debug_info``モジュールの 2 つのモジュールにしか影響しません。

注釈: このドキュメントでは、**"modules"**, **"archives"**, **"base Qt installation"** はそれぞれ別のものを指しており、ここで定義されています:

- **Archives**: このコンテキストでは、**Archive** は 7-zip で圧縮されたファイルのバンドルです。これはディスクドライブ上に拡張子 ``.7z``"のファイルとして存在します。
- **Modules**: Qt リポジトリが一連のアーカイブをモジュールとして編成しています。**module** には、ひとつか複数の **archives** が含まれます。
- **the base Qt installation**: 定義上、これは 20 から 30 の**archives** を含む単なる **module** です。このドキュメントでは、さまざまな事情から**module**ではなく、 **the base Qt installation** とよんでいます。
  - `aqt install-qt` はデフォルトでこのモジュールをインストールします。
  - このモジュールを ``aqt install-qt --modules`` で指定することはできません。
  - `aqt list-qt--modules` コマンドはこのモジュールの説明を表示できません。
  - `aqt` は Qt リポジトリに存在するモジュールの名前を読みやすく書きやすくするために変換します。**the base Qt installation** の名前を同じ規則を使って変換された場合、名前は空になります。

    base Qt installation モジュールの完全修飾名は通常、`qt.qt6.620.gcc_64` のようなものです。`qtcharts` モジュールの完全修飾名は通常、`qt.qt6.620.qtcharts.gcc_64` のようなものです。プレフィックス `qt.qt6.620.``**とサフィックス**.gcc_64``を持つ 20 個のモジュールのリストを読み書きするのは難しいでしょう。なぜならこれらの部分は反復的で意味がないからです。ただ ``qtcharts`` の部分だけが役に立つからです。

---

gcc_64 アーキテクチャを使って Qt 5.15.2 を Linux デスクトップにインストールしたいとしましょう。`qtbase` アーカイブには、動作中の Qt インストールに最低限必要なものが含まれています。また、`--archives` フラグを付けて単独でインストールすることもできます。

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

今回、``aqt install-qt``は、デフォルトでインストールされる約 27 のアーカイブではなく、``qtbase``という 1 つのアーカイブだけをインストールします。

3.8.3 最小インストール数を超えるインストール

qtbase``**アーカイブに必要な機能が欠けているとします。**--archives qtbase``フラグを使用すると、``aqt install-qt``は約 27 のアーカイブを省略します。``aqt list-qt --archives``コマンドでこれらのアーカイブのリストを表示することができます:

```
$ 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
```

ここでは、関心のある Qt のバージョンと使用しているアーキテクチャの 2 つの引数を持つ``--archives``フラグを使用しました。その結果、コマンドはベース (最小ではない)Qt インストールの一部であるアーカイブのリストを出力しました。

qtmultimedia、qtdeclarative、``qtsvg``を使う必要があり、それ以外は何も使わないとしましょう。``qtbase``アーカイブは最低限動作する Qt インストールに必要であることを覚えておいてください。これらのアーカイブは次のコマンドを使ってインストールできます:

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

3.8.4 アーカイブを指定してモジュールをインストールする

aqt v2.1.0 以降では、--archives``**フラグは** Qt **の基本インストールと** ``debug_info``**モジュールにのみ適用されます。**aqt **の以前のバージョンでは、**--archives``フラグを付けてモジュールをインストールする場合、ユーザはモジュールごとにアーカイブを指定する必要がありました。指定しないとインストールされませんでした。このようなミスを防止するために、この動作を変更しました。

モジュール``qtcharts``と``qtlottie``を使って、最低限の Qt 5.15.2 をインストールする必要があるとしましょう。

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

このコマンドは 3 つのアーカイブを正しくインストールします。1 つは``qtbase``用で、もう 1 つは 2 つのモジュール用です。このコマンドを aqt の以前のバージョンで使用しようとすると、2 つのモジュールは``--archives``リストに指定されていなかったので、インストールされませんでした。

注釈: qtbase``**アーカイブを省略したり、他のアーカイブやモジュールが依存しているアーカイブを省略することで、**--archives``フラグを誤用することもできます。プログラムをコンパイルしようとしてコンパイルが失敗するまで、問題があることに気づかないかもしれません。

3.8.5 ``debug_info``モジュールのインストール

ここで、debug_info``**モジュールをインストールする必要があるとしましょう。このモジュールは非常に大きく、約** 1.0GB **です。すべてをインストールしたくないので、**``aqt install-qt --archives``**を使ってインストールしたいアーカイブを選ぶことができます。この場合の**--archives``フラグは

``aqt list-qt --archives``で、どのアーカイブが``debug_info``モジュールの一部であるかを表示します。

```
$ 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
```

ここにはたくさんのアーカイブがあります。``debug_info``アーカイブと他のすべてのモジュール/Qt ベースインストールのアーカイブとの間に名前の衝突があることに注意してください。これは、利用可能な他のほとんどすべてのアーカイブに対応する``debug_info``アーカイブがあるからです。

``qtcharts``と``debug_info``つきで Qt をインストールし、いくつかのアーカイブを指定しましょう。

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

ここで行ったことに注目してください:qtcharts モジュールと``debug_info``モジュールを指定し、qtbase``**アーカイブ、**``qtcharts``**アーカイブ、**``qtdeclarative アーカイブを指定しました。これで合計 6 つのアーカイブがインストールされます:

- debug_info``**モジュールからの** 3 **つのアーカイブ** ``qtbase、qtcharts、qtdeclarative
- qtcharts``**モジュールからのアーカイブ** ``qtcharts、および
- 基本 Qt インストールの``qtbase``と``qtdeclarative``の 2 つのアーカイブです。

注釈: 現在、aqt install-qt``**は、基本** Qt **インストールから対応するモジュールをインストールせずに、**``debug_info モジュールからアーカイブをインストールすることはできません。例えば、通常の``qtbase``アーカイブをインストールせずに、qtbase``**用の** ``debug_info アーカイブをインストールすることはできません。

第 4 章

コマンドラインオプション

コマンドラインのインタフェースは、argparse ライブラリを使用してコマンドラインのオプションを解析します。そのため、オプションに、短いバージョンや長いバージョンが使用できるます。長いオプションは最も短い明確な省略形も提供ます。

4.1 一般的なコマンド

```
aqt help
```

汎用ヘルプの表示

```
aqt version
```

バージョンの表示

4.2 List-*コマンド

このコマンドは、``aqt`でインストールできるパッケージを一覧表示するために使用します。

4.2.1 list-qt コマンド

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

(次のページに続く)

(前のページからの続き)

```
            --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、windows、または mac

target

desktop、winrt、ios または android。省略すると、ホスト OS で使用可能なすべてのターゲットが出力されます。winrt は Windows でしか使用できず、ios は Mac OS でしか使用できません。

--help, -h

ヘルプテキストを表示する

--spec <Specification>

バージョンの範囲を指定する SimpleSpec 内の Qt のバージョンを表示します。部分的なバージョンや不等式などを指定できます。"*" は Qt のすべてのバージョンにマッチし、">6.0.2,<6.2.0" は 6.0.2 から 6.2.0 までの Qt のすべてのバージョンにマッチします。例えば、`aqt list-qt windows desktop --spec "5.12"` と入力すると、5.12 から始まる Windows デスクトップ用の Qt のすべてのバージョンが出力されます。他のフラグと組み合わせて、そのフラグの出力をフィルタリングすることができます。

--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>

モジュールの長い表示: `--modules` と似ていますが、各モジュールに関連付けられた追加のメタデータを表示します。このメタデータは、各モジュールの長い表示名を含む表に表示されます。端末の表示幅が 95 文字を超える場合、`aqt list-qt` には各モジュールのリリース日とサイズも表示されます。この出力例を以下に示します。

```
$ 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    ␣
```

(次のページに続く)

(前のページからの続き)

```
↪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
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
```

(次のページに続く)

(前のページからの続き)

```
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...]

このフラグには、'Qt version' と'architecture' の 2 つ以上の引数のリストが必要です。'Qt version' 引数は、"5.X.Y"または"latest"キーワードの形式にすることができます。'architecture' 引数に指定できる値のリストを見るには、`--arch` フラグを使用します。後続の引数は、以前のバージョンとアーキテクチャーで使用できるモジュールの名前である必要があります。受け入れ可能な値のリストを見るために、`--modules` フラグを使うことができます。

このフラグにモジュールのリストを追加しない場合、このコマンドは基本 Qt インストールを構成するすべてのアーカイブのリストを出力します。

このフラグにモジュールのリストを追加すると、指定されたモジュールを構成するすべてのアーカイブのリストが出力されます。

このコマンドの目的は、 `install-*` コマンドを使うときに *archives flag* に渡すことができる引数を表示することです。このフラグを使うと、Qt の不要な部分をインストールするのを避けることができます。

--latest-version

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

4.2.2 list-src コマンド

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

install-src command を使って、インストール可能なソースアーカイブを一覧表示します。

host

linux、windows、または mac

Qt version

これは 5.9.7、5.12.1 などの Qt バージョンです。有効なバージョンを調べるには、*list-qt* **コマンド** を使用してください。

spec

これはバージョンの範囲を指定する SimpleSpec です。バージョンとして解釈できない何かを位置引数 <Qt version> に入力すると、それは SimpleSpec として解釈され、aqt はその `SimpleSpec`_ 内で利用可能な最高バージョンを選択します。

例えば、``aqt list-src mac 5.12``は入手可能な最新バージョンの Qt 5.12(この記事を書いている時点では 5.12.11) のアーカイブを出力します。

4.2.3 list-doc コマンド

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

install-doc **コマンド** を使ってインストール可能なドキュメントアーカイブとモジュールをリストアップできます。

デフォルトでは、 list-doc は *install-doc* **コマンド** と --archives オプションを使ってインストール可能なアーカイブのリストを出力します。

host

linux、windows、または mac

Qt version

これは 5.9.7、5.12.1 などの Qt バージョンです。有効なバージョンを調べるには、*list-qt* **コマンド** を使用してください。

spec

これはバージョンの範囲を指定する SimpleSpec です。バージョンとして解釈できない何かを位置引数 <Qt version> に入力すると、それは SimpleSpec として解釈され、aqt はその `SimpleSpec`_ 内で利用可能な最高バージョンを選択します。

例えば、aqt list-doc mac 5.12 は入手可能な最新バージョンの Qt 5.12(この記事を書いている時点では 5.12.11) のアーカイブを出力します。

--modules

このフラグは、list-doc``に `install-doc command`_ と ``--modules オプションを使ってインストール可能なモジュールのリストを出力させます。

4.2.4 list-example コマンド

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

install-example command を使ってインストール可能なアーカイブとモジュールの例を挙げます。

デフォルトでは、list-example は *install-example command* と --archives オプションを使ってインストール可能なアーカイブのリストを出力します。

host

linux、windows、または mac

Qt version

これは 5.9.7、5.12.1 などの Qt バージョンです。有効なバージョンを調べるには、*list-qt* **コマンド** を使用してください。

spec

これはバージョンの範囲を指定する SimpleSpec です。バージョンとして解釈できない何かを位置引数 <Qt version> に入力すると、それは SimpleSpec として解釈され、aqt はその `SimpleSpec`_ 内で利用可能な最高バージョンを選択します。

例えば、 aqt list-example mac 5.12 は入手可能な最新バージョンの Qt 5.12(この記事を書いている時点では 5.12.11) のアーカイブを出力します。

--modules

このフラグは、 list-example に *install-example command* と --modules オプションを使ってインストール可能なモジュールのリストを出力させます。

4.2.5 list-tool コマンド

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

使用可能なツールを一覧表示

host

linux、windows、または mac

target

desktop、winrt、ios または android。省略すると、ホスト OS で使用可能なすべてのターゲットが出力されます。winrt は Windows でしか使用できず、ios は Mac OS でしか使用できません。

tool name

ツールの名前です。利用可能な値を確認するには、 `aqt list-tool <host> <target>` を使用してください設定すると、利用可能なすべての ' ツールバリアント名' を出力します。

このコマンドの出力は、以下の *aqt install-tool* で使うことを意図しています。

--help, -h

ヘルプテキストを表示する

--long, -l

長い表示: 各ツールバリアントに関連付けられた追加のメタデータを表示します。このメタデータは表形式で表示され、各ツールのバージョンとリリース日が含まれます。あなたの端末の表示幅が 95 文字を超える場合、`aqt list-tool` は各ツールの名前と説明も表示します。この出力例を以下に示します。

```
$ 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-*コマンド

これらのコマンドは、Qt、ツール、ソース、ドキュメント、サンプルをインストールするために使用されます。

4.3.1 共通オプション

これらのコマンドのほとんどは、同じコマンドラインオプションを共有しています。次に、これらのオプションについて説明します。

--help, -h

ヘルプテキストを表示する

--outputdir, -O <Output Directory>

出力ディレクトリを指定します。デフォルトでは、aqt は現在の作業ディレクトリにインストールされます。

--base, -b <base url>

'online' フォルダが存在する -b `https://mirrors.dotsrc.org/qtproject` などのミラーサイトベース URL を指定します。

--config, -c <settings_file_path>

独自の `settings.ini` ファイルへのパスを指定します。*the Configuration section* を参照してください。

--timeout <timeout(sec)>

ダウンロードサイトの接続タイムアウト (秒単位)。(デフォルト:5 秒)

--external, -E <7zip command>

外部 7zip コマンドのパスを指定します。デフォルトでは、aqt はこのタスクに py7zr を使用します。

これまで、ユーザは Windows、Linux、Mac で 7-zip を使ってうまくやってきました。Windows では Choco を使って 7-zip をインストールすることができます。Linux/Mac のポートである 7-zip は `p7zip` と呼ばれており、brew on Mac でインストールするか、Linux ではパッケージマネージャを使ってインストールすることができます。

--internal

内部エクストラクタ py7zr を使用します。

--keep, -k

指定された場合はダウンロードされたアーカイブを保持します。指定されていない場合はインストール後に削除します。これらのファイルを aqt が配置する場所を選択するには、``--archive-dest <path>``を使用します。ダウンロード先を指定しない場合、aqt はこれらのファイルを現在の作業ディレクトリに配置します。

--archive-dest <path>

ダウンロードしたアーカイブの保存先パスを設定します (デフォルトディレクトリ) を設定します。上記の --keep オプションが指定されていないか、または aqt がクラッシュしない限り、ダウンロードしたアーカイブはすべて自動的に削除されます。

このオプションは、 aqt がダウンロードして --outputdir に展開する中間の .7z アーカイブを参照していることに注意してください。ほとんどのユーザはこれらのファイルを保持する必要はありません。

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

リストとしてインストールする追加モジュールを指定してください。適切な aqt list-* コマンドを使って、利用可能なモジュールをリストしてください:

インストール・コマンド	リストコマンド	リストコマンドの使用法
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

このオプションは、 install-qt、 install-example 、 install-doc にのみ適用できます。

次のように、複数のモジュールをインストールできます。

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

利用可能なすべてのモジュールをインストールしたい場合は、次のように、モジュールのリストの代わりに ``all``キーワードを使うことができます。

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

--archives <list of archives>

[高度] アーカイブのサブセットを指定して、インストールされているアーカイブを**制限**します。これは Qt の基本インストールと debug_info モジュールにのみ影響します。これは高度なオプションであり、一般的な使用にはお勧めできません。主な目的は、インストールされているモジュールを制限することによって CI/CD プロセスを高速化することです。Qt SDK のインストールが失敗する可能性があります。

このオプションは、install-tool``**を除くすべての** ``install-* コマンドに適用できます。

適切な ``aqt list-*`` コマンドを使えば、このコマンドで使える全ての値のリストを出力することができます。

インストール・コマンド	リストコマンド	リストコマンドの使用法
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 コマンド

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

指定されたバージョンとターゲットで Qt ライブラリをインストールします。Qt のバージョンに応じて、さまざまな組み合わせを受け入れることができます。

host

linux, windows または、mac. Qt 開発ツールが実行されるオペレーティングシステム。

target

desktop, ios, winrt, または androidQt プログラムを開発するデバイスのタイプ。ターゲットが ios の場合、6.2.4 よりも古いバージョンの Qt は、現在のバージョンの XCode では機能しないことに注意してください (バージョン 13 以上の XCode に適用されます)。

Qt version

これは 5.9.7、5.12.1 などの Qt バージョンです。有効なバージョンを調べるには、*list-qt* **コマンド** を使用してください。

spec

これはバージョンの範囲を指定する SimpleSpec です。バージョンとして解釈できない何かを位置引数 <Qt version> に入力すると、それは SimpleSpec として解釈され、aqt はその `SimpleSpec`_ 内で利用可能な最高バージョンを選択します。

例えば、aqt install-qt mac desktop 5.12 は Qt 5.12 の最新バージョンをインストールし、aqt install-qt mac desktop "*" は Qt の最新バージョンをインストールします。

このオプションを使用すると、aqt はインストールされたバージョンをログに出力するので、簡単に確認できます。

arch

開発対象のコンパイラアーキテクチャ。オプション:

- Linux デスクトップの場合： gcc_64
- mac デスクトップの場合： clang_64
- Windows デスクトップの場合は、win64_msvc2019_64, win64_msvc2017_64, win64_msvc2015_64, win32_msvc2015, win32_mingw53
- Android の場合は： android_armv7, android_arm64_v8a, android_x86, android_x86_64

list-qt **コマンド** を使用して、利用可能なアーキテクチャを一覧表示してください。

--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

[高度] すべての基本パッケージをインストールしないように指定します。これは拡張オプションなので、--modules オプションと一緒に使うべきです。これにより、既存の Qt インストールにモジュールを追加できます。

common options 参照

4.3.3 install-src コマンド

```
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>)
```

指定されたバージョンとターゲットの Qt ソースコードをインストールします。

host

linux、windows、または mac

target

将来のバージョンの aqt では廃止され、削除される予定です。このパラメータは下位互換性のために存在し、その値は無視されます。

Qt version

これは 5.9.7、5.12.1 などの Qt バージョンです。有効なバージョンを調べるには、*list-qt* **コマンド** を使用してください。

spec

これはバージョンの範囲を指定する SimpleSpec です。バージョンとして解釈できない何かを位置引数 <Qt version> に入力すると、それは SimpleSpec として解釈され、aqt はその `SimpleSpec`_ 内で利用可能な最高バージョンを選択します。

例えば、`aqt install-src mac 5.12` は入手可能な最新バージョンの Qt 5.12 のソースをインストールし、`aqt install-src mac "*"` は入手可能な最新バージョンの Qt のソースをインストールします。

--kde

「--kde」オプションを追加すると、qtbase のソースツリーに KDE パッチコレクションが適用されます。これは Qt 5.15.2 にのみ適用されます。指定されたバージョンがそれ以外の場合、--kde を指定したコマンドはエラーで中断されます。

common options 参照

4.3.4 install-doc コマンド

```
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>)
```

指定されたバージョンとターゲットの Qt ドキュメントをインストールします。

host

linux、windows、または mac

target

将来のバージョンの aqt では廃止され、削除される予定です。このパラメータは下位互換性のために存在し、その値は無視されます。

Qt version

これは 5.9.7、5.12.1 などの Qt バージョンです。有効なバージョンを調べるには、*list-qt* **コマンド** を使用してください。

spec

これはバージョンの範囲を指定する SimpleSpec です。バージョンとして解釈できない何かを位置引数 <Qt version> に入力すると、それは SimpleSpec として解釈され、aqt はその `SimpleSpec`_ 内で利用可能な最高バージョンを選択します。

例えば、aqt install-doc mac 5.12 と指定すると Qt 5.12 の最新バージョンのドキュメントがインストールされ、aqt install-doc mac "*" と指定すると Qt の最新バージョンのドキュメントがインストールされます。

common options 参照

4.3.5 install-example コマンド

```
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>)
```

指定されたバージョンとターゲットの Qt サンプルをインストールします。

host

linux、windows、または mac

target

将来のバージョンの aqt では廃止され、削除される予定です。このパラメータは下位互換性のために存在し、その値は無視されます。

Qt version

これは 5.9.7、5.12.1 などの Qt バージョンです。有効なバージョンを調べるには、*list-qt* **コマンド** を使用してください。

spec

これはバージョンの範囲を指定する SimpleSpec です。バージョンとして解釈できない何かを位置引数 <Qt version> に入力すると、それは SimpleSpec として解釈され、aqt はその `SimpleSpec`_ 内で利用可能な最高バージョンを選択します。

例えば、``aqt install-example mac 5.12``は Qt 5.12 の最新バージョンのサンプルをインストールし、``aqt install-example mac "*"``は Qt の最新バージョンのサンプルをインストールします。

common options 参照

4.3.6 install-tool コマンド

```
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>]
```

QtIFW、mingw、Cmake、Conan、vcredist などのツールをインストールします。

host

linux、windows、または mac

target

desktop、ios、または Android

tool name

指定されたツールをインストールします。ツール名は'tools_openssl_x64'、'tools_vcredist'、'tools_ninja'、'tools_ifw'、'tools_cmake' のいずれかです。

tool variant name

ツールバリアントを指定するオプションのフィールド。vcredist および mingw のインストールで必要になる場合があります。ツールバリアント名は'qt.tools.win64_mingw810','qt.tools.vcredist_msvc2013_x64' のようになります。

:ref:`List-Tool command`を使用して、使用可能なツールとツールバリアント名を表示する必要があります。

common options 参照

4.4 従来のサブコマンド

サブコマンドの「install」、「tool」、「src」、「doc」、および「examples」は、新しい「install-*」コマンドのために廃止されましたが、まだ使用する必要がある場合に備えて、aqt に残っています。これらの古いコマンドのドキュメントは、https://aqtinstall.readthedocs.io/en/v1.2.4/でまだ入手できます。

第 5 章

コマンドの例

例:QtCharts と QtNetworkAuth で Qt SDK 5.12.12 for Linux をインストールする:

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

例:Qt 5.12 の最新の LTS バージョンをインストールする

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

例:Android(armv7)Qt 5.13.2 のインストール:

```
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
```

例: example、doc および source のインストール:

```
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
```

例: `install-example/doc/src` でインストール可能なアーカイブを一覧表示します。

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

例: install-example/doc でインストール可能なモジュールを一覧表示します:

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

例: Web アセンブリのインストール

```
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
```

例: Linux で利用可能な Qt のバージョンを一覧表示する

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

例:macOS 上で利用可能な Qt6 のバージョンを一覧表示する

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

例:macOS 上の Qt の最新バージョンで利用可能なモジュールを一覧表示する

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

例:Windows 上の Qt 6.1.2 で利用可能なアーキテクチャを一覧表示する

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

例: WIndows で使用可能なツールを一覧表示する

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

例: 使用可能な IFW のバリエーションを一覧表示する。

```
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'
```

例: バージョン、リリース日、説明など、IFW のバリエーションを一覧表示する。

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

例:Install FrameWork(IFW) のインストール:

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

例:Windows への vcredist のインストール:

```
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
```

注釈: 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.

例: ヘルプメッセージの表示

```
aqt help
```

第 6 章

構成

aqtinstall``は設定ファイルで設定できます。デフォルトの設定は ``aqt/settings.ini ファイルに保存されています。

環境変数 AQT_CONFIG やコマンドラインオプション``-c``や``--config`` でカスタム設定ファイルを指定することができます。

設定ファイルは次のようになります。

```
[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

```

(次のページに続く)

(前のページからの続き)

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

6.1 設定

[aqt] セクションは基本的な動作を設定します。

concurrency:
: concurrency は、同時に開始するダウンロードの数を設定します。これは整数値でなくてはなりません。

baseurl:
: baseurl は Qt ダウンロードサイトの URL です。独自の Qt ダウンロードサイトリポジトリがある場合は、ここで設定できます。これは``--base``オプションと同じです。

7zcmd:
: 圧縮解凍ツールの 7-zip のコマンド名です。aqtinstall が **推奨ライブラリ ``py7zr``なしで** インストールされている場合、py7zr ライブラリの代わりにアーカイブを抽出するために使用されます。--external オプションが指定されている場合、オプションの指定値で上書きされます。

print_stacktrace_on_error:
: print_stacktrace_on_error は、 True か ``False``を値にとります。True に設定すると、プログラムを終了させるエラーが発生するたびにスタックトレースが stderr に出力されます。``False``に設定すると、未処理の例外が発生しない限り、スタックトレースは表示されません。

always_keep_archives:
: これは、 True``**か** ``False を値にとります。True が設定さえると、 --keep オプションが、aqt 実行時に毎回指定されたことになります。コマンドラインオプションで上書きできません。 False が設定されると、

aqt 実行するときに ``.7z``アーカイブを削除したくないときは、`--keep` を手動で設定する必要があります。

archive_download_location: これは、`--keep` がオンになっている場合に、`.7z` アーカイブがダウンロードされる場所への相対パスまたは絶対パスです。この場所は、`--archives-dest` オプションで上書きできます。

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.

この設定はこれらのモジュールをインストールする能力に影響を与えません。`aqt install-*` は警告なしでそれらをインストールします。

`[requests]` セクションは、`aqt` がネットワークリクエストを行う方法を制御します。

connection_timeout:

`connection_timeout` は接続の秒単位のタイムアウトです。これは ``requests``ライブラリに渡されます。

response_timeout:

``response_timeout``は応答待ち時間の秒単位のタイムアウトです。これは ``requests``ライブラリに渡されます。

max_retries: 廃止されました。この設定は使用しないでください。

max_retries_on_connection_error:

`max_retries_on_connection_error` は、接続エラーが発生した場合に `aqt` がサーバに再接続しようとする回数を制御する整数です。

retry_backoff:

`retry_backoff` は浮動小数点数で、`aqt` が失敗した接続試行間にスリープする時間を制御します。この値をあまりに低く設定するとサーバに衝撃を与え、接続が全く成功しなくなる可能性があります。

max_retries_on_checksum_error: この設定は、チェックサムエラーが発生した場合に `aqt` がファイルをダウンロードしようとする回数を制御します。

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.

`[mirrors]` セクションはミラー処理のための設定です。

trusted_mirrors:

`trusted_mirrors` は、ダウンロードされたすべてのアーカイブに対して正確なチェックサムを提供するために信頼している URL のリストです。これはセキュリティ機能です。何をしているのかわからない限り、この値を変更しないでください!

`aqtinstall` は、このリスト内のミラーからすべてのチェックサムをダウンロードします。これらのチェックサムは、`aqtinstall` がダウンロードする他のすべてのファイルが、実際には正しいファイルであり、そのファイルの不正コピーや悪意のあるコピーではないことを確認するために使用されます。デフォルトのミラーにアクセスできない場合や、ミラーが侵害されていないと信頼できない場合は、このリストを変更する必要があります。

`aqtinstall` は、フォールバックミラーリストからアーカイブファイルを安全にダウンロードし、`trusted_mirrors` リストからダウンロードしたチェックサムに照らして、それらが悪意のあるファイルでないことを確認することができます。`aqtinstall` は、SHA-256 アルゴリズムを使ってこのチェックを行います。

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: ア

クセスに適した URL の一覧です。ミラーサイトでエラーが発生した場合は、可能な場合はフォールバックを使用してください。ミラーの一覧は次の場所で確認できます: https://download.qt.io/static/mirrorlist/

第 7 章

コントリビューションガイド

aqtinstall プロジェクトへの貢献ガイドです。Pull-Request を送信してバグを報告し、質問をすることを歓迎します。

7.1 リソース

- プロジェクトオーナー: 三浦広志
- バグトラッカー: Github issue Tracker
- ステータス: ベータ
- 活動度: 中等度

7.2 バグ・トリアージ

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

7.3 パッチを送信

プロジェクトにパッチを送りたい場合の少量のルールを以下に示します。

1. 修正案はすべて「プルリクエスト」として送信する必要があります。
1. 各プルリクエストは複数のコミットで構成されます。
1. 変更を論理的なサブパートである個々のコミットに分割することをお勧めします。

7.4 CI テスト

プロジェクトは、Azure Pipelines、Github アクション、Coveralls を使用してリグレッションテストを行うように設定されています。テスト結果はバッジで確認でき、詳細はバッジからリンクされた Web ページで確認できます。

第 8 章

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.

第 9 章

セキュリティポリシー

9.1 サポートされているバージョン

バージョン	ステータス
3.1.x	安定バージョン
3.0.x	セキュリティ修正のみ
< 3.0	サポートされない

9.2 脆弱性の報告

セキュリティの脆弱性については、miurahr@linux.com に提示してください。

第 10 章

作者

Aqtinstall は、三浦広志 <miurahr@linux.com> 氏により作成され、管理されています。

オリジナルの qli-installer は Linus Jahn 氏によって書かれました。

David Dalcino によるバージョン 2.0 と 2.1 の改善への重要な貢献がありました。バージョン 2.0 以降の多くの開発とレビューの取り組みも主導しています。

すべてのコントリビュータ (アルファベット順) は次のとおりです。

- 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)

他にも多くの参加者や貢献者がいます。記録に名前がない方は、遠慮なく言ってください。

第 11 章

ChangeLog

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

11.1 Unreleased

11.2 v3.1.11 (28, Nov. 2023)

11.2.1 Fixed

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

11.2.2 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

11.3 v3.1.10 (14, Nov. 2023)

11.3.1 Fixed

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

11.3.2 Changed

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

11.4 v3.1.9 (6, Nov. 2023)

11.4.1 Security

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

11.4.2 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)

11.4.3 Added

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

11.5 v3.1.8 (1, Nov. 2023)

11.5.1 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)

11.5.2 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)

11.6 v3.1.7 (1, Aug. 2023)

11.6.1 Added

Add support for standalone sdktool installation(#677)

11.6.2 Fixed

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

11.6.3 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)

11.7 v3.1.6 (4, May, 2023)

11.7.1 Added

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

11.8 v3.1.5 (30, Mar. 2023)

11.8.1 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)

11.9 v3.1.4 (25, Mar. 2023)

11.9.1 Changed

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

11.10 v3.1.3 (2, Mar. 2023)

11.10.1 Changed

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

11.11 v3.1.2 (17, Feb. 2023)

11.11.1 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)

11.11.2 Changed

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

11.12 v3.1.1 (10, Feb. 2023)

11.12.1 Fixed

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

11.13 v3.1.0 (5, Dec. 2022)

11.13.1 Fixed

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

11.13.2 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

11.13.3 Deprecated

- Drop support for python 3.6

11.14 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)

11.15 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

11.16 v3.0.0 (29, Sep. 2022)

11.16.1 Added

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

11.16.2 Fixed

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

11.16.3 Changed

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

11.17 v2.2.3 (17, Aug. 2022)

11.17.1 Fixed

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

11.17.2 Added

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

11.18 v2.2.2 (11, Aug. 2022)

11.18.1 Added

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

11.18.2 Fixed

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

11.18.3 v2.2.1 (9, Aug. 2022)

11.18.4 Changed

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

11.19 v2.2.0 (2, Aug. 2022)

11.19.1 Added

- Add code of conduct (#535)

11.19.2 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)

11.19.3 Fixed

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

11.19.4 Deprecated

- Deprecate setup.py file (#531)

11.20 v2.1.0 (14, Apr. 2022)

11.20.1 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)

11.20.2 Fixed

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

11.20.3 Added

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

11.20.4 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)

第 12 章

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

第 13 章

Indices and tables

- genindex
- modindex
- search

索引