

NAME

cmake-qt – CMake Qt Features Reference

INTRODUCTION

CMake can find and use Qt 4, Qt 5 and Qt 6 libraries. The Qt 4 libraries are found by the *FindQt4* find-module shipped with CMake, whereas the Qt 5 and Qt 6 libraries are found using "Config-file Packages" shipped with Qt 5 and Qt 6. See *cmake(7)* for more information about CMake packages, and see *the Qt cmake manual* for your Qt version.

Qt 4, Qt 5 and Qt 6 may be used together in the same *CMake buildsystem*:

```
cmake_minimum_required(VERSION 3.16 FATAL_ERROR)

project(Qt4_5_6)

set(CMAKE_AUTOMOC ON)

find_package(Qt6 COMPONENTS Widgets DBus REQUIRED)
add_executable(publisher publisher.cpp)
target_link_libraries(publisher Qt6::Widgets Qt6::DBus)

find_package(Qt5 COMPONENTS Gui DBus REQUIRED)
add_executable(subscriber1 subscriber1.cpp)
target_link_libraries(subscriber1 Qt5::Gui Qt5::DBus)

find_package(Qt4 REQUIRED)
add_executable(subscriber2 subscriber2.cpp)
target_link_libraries(subscriber2 Qt4::QtGui Qt4::QtDBus)
```

A CMake target may not link to more than one Qt version. A diagnostic is issued if this is attempted or results from transitive target dependency evaluation.

QT BUILD TOOLS

Qt relies on some bundled tools for code generation, such as **moc** for meta-object code generation, **uic** for widget layout and population, and **rcc** for virtual file system content generation. These tools may be automatically invoked by *cmake(1)* if the appropriate conditions are met. The automatic tool invocation may be used with Qt version 4 to 6.

AUTOMOC

The *AUTOMOC* target property controls whether *cmake(1)* inspects the C++ files in the target to determine if they require **moc** to be run, and to create rules to execute **moc** at the appropriate time.

If a macro from *AUTOMOC_MACRO_NAMES* is found in a header file, **moc** will be run on the file. The result will be put into a file named according to **moc_<basename>.cpp**. If the macro is found in a C++ implementation file, the moc output will be put into a file named according to **<basename>.moc**, following the Qt conventions. The **<basename>.moc** must be included by the user in the C++ implementation file with a preprocessor **#include**.

Included **moc_*.cpp** and ***.moc** files will be generated in the **<AUTOGEN_BUILD_DIR>/include** directory which is automatically added to the target's *INCLUDE_DIRECTORIES*.

- This differs from CMake 3.7 and below; see their documentation for details.
- For *multi configuration generators*, the include directory is **<AUTOGEN_BUILD_DIR>/include_<CONFIG>**.

- See `AUTOGEN_BUILD_DIR`.

Not included `moc_<basename>.cpp` files will be generated in custom folders to avoid name collisions and included in a separate file which is compiled into the target, named either `<AUTOGEN_BUILD_DIR>/mocs_compilation.cpp` or `<AUTOGEN_BUILD_DIR>/mocs_compilation_${<CONFIG>.cpp`.

- See `AUTOGEN_BUILD_DIR`.

The `moc` command line will consume the `COMPILE_DEFINITIONS` and `INCLUDE_DIRECTORIES` target properties from the target it is being invoked for, and for the appropriate build configuration.

The `AUTOMOC` target property may be pre-set for all following targets by setting the `CMAKE_AUTOMOC` variable. The `AUTOMOC_MOC_OPTIONS` target property may be populated to set options to pass to `moc`. The `CMAKE_AUTOMOC_MOC_OPTIONS` variable may be populated to pre-set the options for all following targets.

Additional macro names to search for can be added to `AUTOMOC_MACRO_NAMES`.

Additional `moc` dependency file names can be extracted from source code by using `AUTOMOC_DEPEND_FILTERS`.

Source C++ files can be excluded from `AUTOMOC` processing by enabling `SKIP_AUTOMOC` or the broader `SKIP_AUTOGEN`.

AUTOUIC

The `AUTOUIC` target property controls whether `cmake(1)` inspects the C++ files in the target to determine if they require `uic` to be run, and to create rules to execute `uic` at the appropriate time.

If a preprocessor `#include` directive is found which matches `<path>ui_<basename>.h`, and a `<basename>.ui` file exists, then `uic` will be executed to generate the appropriate file. The `<basename>.ui` file is searched for in the following places

1. `<source_dir>/<basename>.ui`
2. `<source_dir>/<path><basename>.ui`
3. `<AUTOUIC_SEARCH_PATHS>/<basename>.ui`
4. `<AUTOUIC_SEARCH_PATHS>/<path><basename>.ui`

where `<source_dir>` is the directory of the C++ file and `AUTOUIC_SEARCH_PATHS` is a list of additional search paths.

The generated `ui_*.h` files are placed in the `<AUTOGEN_BUILD_DIR>/include` directory which is automatically added to the target's `INCLUDE_DIRECTORIES`.

- This differs from CMake 3.7 and below; see their documentation for details.
- For *multi configuration generators*, the include directory is `<AUTOGEN_BUILD_DIR>/include_<CONFIG>`.
- See `AUTOGEN_BUILD_DIR`.

The `AUTOUIC` target property may be pre-set for all following targets by setting the `CMAKE_AUTOUIC` variable. The `AUTOUIC_OPTIONS` target property may be populated to set options to pass to `uic`. The `CMAKE_AUTOUIC_OPTIONS` variable may be populated to pre-set the options for all following targets. The `AUTOUIC_OPTIONS` source file property may be set on the `<basename>.ui` file to set particular options for the file. This overrides options from the `AUTOUIC_OPTIONS` target property.

A target may populate the *INTERFACE_AUTOUIC_OPTIONS* target property with options that should be used when invoking **uic**. This must be consistent with the *AUTOUIC_OPTIONS* target property content of the depender target. The *CMAKE_DEBUG_TARGET_PROPERTIES* variable may be used to track the origin target of such *INTERFACE_AUTOUIC_OPTIONS*. This means that a library which provides an alternative translation system for Qt may specify options which should be used when running **uic**:

```
add_library(KI18n klocalizedstring.cpp)
target_link_libraries(KI18n Qt6::Core)

# KI18n uses the tr2i18n() function instead of tr(). That function is
# declared in the klocalizedstring.h header.
set(autouic_options
    -tr tr2i18n
    -include klocalizedstring.h
)

set_property(TARGET KI18n APPEND PROPERTY
    INTERFACE_AUTOUIC_OPTIONS ${autouic_options}
)
```

A consuming project linking to the target exported from upstream automatically uses appropriate options when **uic** is run by *AUTOUIC*, as a result of linking with the *IMPORTED* target:

```
set(CMAKE_AUTOUIC ON)
# Uses a libwidget.ui file:
add_library(LibWidget libwidget.cpp)
target_link_libraries(LibWidget
    KF5::KI18n
    Qt5::Widgets
)
```

Source files can be excluded from *AUTOUIC* processing by enabling *SKIP_AUTOUIC* or the broader *SKIP_AUTOGEN*.

AUTORCC

The *AUTORCC* target property controls whether *cmake(1)* creates rules to execute **rcc** at the appropriate time on source files which have the suffix **.qrc**.

```
add_executable(myexe main.cpp resource_file.qrc)
```

The *AUTORCC* target property may be pre-set for all following targets by setting the *CMAKE_AUTORCC* variable. The *AUTORCC_OPTIONS* target property may be populated to set options to pass to **rcc**. The *CMAKE_AUTORCC_OPTIONS* variable may be populated to pre-set the options for all following targets. The *AUTORCC_OPTIONS* source file property may be set on the **<name>.qrc** file to set particular options for the file. This overrides options from the *AUTORCC_OPTIONS* target property.

Source files can be excluded from *AUTORCC* processing by enabling *SKIP_AUTORCC* or the broader *SKIP_AUTOGEN*.

THE <ORIGIN>_AUTOGEN TARGET

The **moc** and **uic** tools are executed as part of a synthesized *The <ORIGIN>_autogen target custom target* generated by CMake. By default that *The <ORIGIN>_autogen target* inherits the dependencies of the **<ORIGIN>** target (see *AUTOGEN_ORIGIN_DEPENDS*). Target dependencies may be added to the *The <ORIGIN>_autogen target* by adding them to the *AUTOGEN_TARGET_DEPENDS* target property.

NOTE:

If Qt 5.15 or later is used and the generator is either *Ninja* or *Makefile Generators*, see *The <ORIGIN>_autogen_timestamp_deps target*.

THE <ORIGIN>_AUTOGEN_TIMESTAMP_DEPS TARGET

If Qt 5.15 or later is used and the generator is either *Ninja* or *Makefile Generators*, the **<ORIGIN>_autogen_timestamp_deps** target is also created in addition to the *The <ORIGIN>_autogen target* target. This target does not have any sources or commands to execute, but it has dependencies that were previously inherited by the pre-Qt 5.15 *The <ORIGIN>_autogen target* target. These dependencies will serve as a list of order-only dependencies for the custom command, without forcing the custom command to re-execute.

VISUAL STUDIO GENERATORS

When using the *Visual Studio generators*, CMake generates a **PRE_BUILD** custom command instead of the *The <ORIGIN>_autogen target custom target* (for *AUTOMOC* and *AUTOUIC*). This isn't always possible though and an *The <ORIGIN>_autogen target custom target* is used, when either

- the **<ORIGIN>** target depends on *GENERATED* files which aren't excluded from *AUTOMOC* and *AUTOUIC* by *SKIP_AUTOMOC*, *SKIP_AUTOUIC*, *SKIP_AUTOGEN* or *CMP0071*
- *AUTOGEN_TARGET_DEPENDS* lists a source file
- *CMAKE_GLOBAL_AUTOGEN_TARGET* is enabled

QTMAIN.LIB ON WINDOWS

The Qt 4 and 5 *IMPORTED* targets for the QtGui libraries specify that the qtmain.lib static library shipped with Qt will be linked by all dependent executables which have the *WIN32_EXECUTABLE* enabled.

To disable this behavior, enable the **Qt5_NO_LINK_QTMAIN** target property for Qt 5 based targets or **QT4_NO_LINK_QTMAIN** target property for Qt 4 based targets.

```
add_executable(myexe WIN32 main.cpp)
target_link_libraries(myexe Qt4::QtGui)

add_executable(myexe_no_qtmain WIN32 main_no_qtmain.cpp)
set_property(TARGET main_no_qtmain PROPERTY QT4_NO_LINK_QTMAIN ON)
target_link_libraries(main_no_qtmain Qt4::QtGui)
```

COPYRIGHT

2000-2025 Kitware, Inc. and Contributors