

兼容 Qt 5 和 Qt 6

Qt 5 和 Qt 6 中 CMake API 的语义在很大程度上是兼容的。但是，直到 Qt 5.14，所有导入的 Qt 库目标和命令都包含版本号作为名称的一部分。这使得编写应该同时适用于 Qt 5 和 Qt 6 的 CMake 代码变得有些麻烦。因此，Qt 5.15 引入了无版本目标和命令，以便能够编写与不同 Qt 版本基本无关的 CMake 代码。

无版本目标

除了现有的导入目标外，Qt 5.15 还引入了无版本目标。也就是说，要链接到 **Qt 核心**，可以引用，或者：
Qt6::Qt::

```
find_package(Qt6 COMPONENTS Core)
if (NOT Qt6_FOUND)
    find_package(Qt5 5.15 REQUIRED COMPONENTS Core)
endif()

add_executable(helloworld
    ...
)

target_link_libraries(helloworld PRIVATE Qt::)
```

上面的代码片段首先尝试查找 Qt 6 安装。如果失败，它将尝试查找 Qt 5.15 包。无论使用 Qt 6 还是 Qt 5，我们都可以使用导入的目标。Qt::

默认情况下定义无版本目标。在第一次调用之前设置 QT_NO_CREATE_VERSIONLESS_TARGETS 以禁用它们。
find_package()

注意： 导入的 Qt:: 目标将不具有 Qt6:: 目标中可用的目标属性。

无版本命令

从 Qt 5.15 开始，Qt 模块还提供了其命令的无版本变体。例如，您现在可以使用 `qt_add_translation` 来编译翻译文件，而不管您使用的是 Qt 5 还是 Qt 6。

在第一次调用之前设置 QT_NO_CREATE_VERSIONLESS_FUNCTIONS，以防止创建无版本命令。
find_package()

混合 Qt 5 和 Qt 6

在这样的设置中，无版本目标和命令将隐式引用通过找到的第一个 Qt 版本。在第一次调用之前设置 `QT_DEFAULT_MAJOR_VERSION` CMake 变量，以使版本显式。 `find_package`

支持较旧的 Qt 5 版本

如果你还需要支持早于 Qt 5.15 的 Qt 5 版本，你可以通过将当前版本存储在 CMake 变量中来实现：

```
find_package(QT NAMES Qt6 Qt5 REQUIRED COMPONENTS Core)
find_package(Qt${QT_VERSION_MAJOR} REQUIRED COMPONENTS Core)

add_executable(helloworld
    ...
)

target_link_libraries(helloworld PRIVATE Qt${QT_VERSION_MAJOR}::Core)
```

在这里，我们尝试找到第一个 Qt 6，如果 Qt 5 失败，则以名称 `Qt5`。如果找到其中任何一个，则成功，并且 CMake 变量将定义为 `Qt5`。

`find_package(<PackageName>...)` 找到 `Qt5` 版本。

然后，我们通过动态创建名称来再次加载已确定的 Qt 版本的包。这是必需的，因为期望包名称为 `Qt5`，否则将打印错误。 `Qt${QT_VERSION_MAJOR}CMAKE_AUTOMOCQt5Qt6`

我们可以使用相同的模式来指定导入库的名称。在调用之前，CMake 将解析为 `Qt5`。

`target_link_libraries Qt${QT_VERSION_MAJOR}::Widgets Qt5::Widgets Qt6::Widgets`

建议的做法

尽可能使用 CMake 命令的无版本变体。

无版本导入的目标对于需要同时使用 Qt 5 和 Qt 6 进行编译的项目非常有用。由于缺少目标属性，因此不建议默认使用它们。

如果您需要支持早于 Qt 5.15 的 Qt 5 版本，或者如果无法控制 CMake 代码是在可能定义了 `QT_NO_CREATE_VERSIONLESS_FUNCTIONS` 或 `QT_NO_CREATE_VERSIONLESS_TARGETS` 的上下文中加载的，请使用 CMake 命令和目标的版本化版本。在这种情况下，您仍可以通过变量确定实际命令或目标名称来简化代码。

视窗中的统一码支持

在 Qt 6 中，默认情况下，为链接到 Qt 模块的目标设置和编译器定义。这与 qmake 行为一致，但与 Qt 5 中的 CMake API 行为相比，这是一个变化。 `UNICODE_UNICODE`

Call `qt_disable_unicode_defines()` on the target to not set the definitions.

```
find_package(Qt6 COMPONENTS Core)

add_executable(helloworld
    ...
)
```

© 2022 The Qt Company Ltd. Documentation contributions included herein are the copyrights of their respective owners. The documentation provided herein is licensed under the terms of the [GNU Free Documentation License version 1.3](#) as published by the Free Software Foundation. Qt and respective logos are [trademarks](#) of The Qt Company Ltd. in Finland and/or other countries worldwide. All other trademarks are property of their respective owners.



Contact Us

Company

- About Us
- Investors
- Newsroom
- Careers
- Office Locations

Licensing

- Terms & Conditions
- Open Source
- FAQ

Support

- Support Services
- Professional Services
- Partners
- Training

For Customers

- Support Center
- Downloads
- Qt Login
- Contact Us
- Customer Success

Community

- Contribute to Qt
- Forum
- Wiki
- Downloads
- Marketplace

