

Qt 6.4 > Build with CMake > [Building projects on the command line](#)

Building projects on the command line

This page explains how to configure and build existing projects. If you want to know how to create a Qt-based CMake project, see the documentation on [how to get started with CMake](#).

To build a Qt project, CMake needs to know where the Qt installation is located. Usually this is done by setting the CMake variable `CMAKE_PREFIX_PATH` to Qt's installation prefix. If you are cross-compiling, see [Cross-compiling](#) for details on additional variables you will need to set.

If Qt is installed using the online installer, choose a Qt version within the top-level installation directory. For example, the following command shows how this is done on Windows:

```
cmake -DCMAKE_PREFIX_PATH=C:\Qt\6.4.0\msvc2019_64 -S <source-dir> -B <build-dir>
```

The `<source-dir>` and `<build-dir>` placeholders represent the source and build directories of your project.

CMake generators

CMake generates the necessary build system files that enable build tools such as GNU Make or Ninja to build your project.

CMake's default generator depends on the platform and build environment. For example on Windows, CMake generates Visual Studio project files if a Visual Studio environment is detected.

For a consistent developer experience on all platforms, use the `generator.Ninja` or `generator.Multi-Config`.

You can select the CMake generator either by setting the environment variable or using the argument: `CMAKE_GENERATOR=Ninja`

```
cmake -G Ninja ...
```

qt-cmake

The script is a convenient alternative to configure your project. It eliminates the need for you to specify the `CMAKE_PREFIX_PATH`. You can find it located in the `bin` directory of your Qt installation prefix. The script passes all parameters to CMake, so you can use it just like you would `cmake`:

```
C:\Qt\6.4.0\msvc2019_64\bin\qt-cmake -G Ninja -S <source-dir> -B <build-dir>
```

After the build system files are generated, your project is ready to be built:

```
cd <build-dir>
ninja
```

```
cmake --build <build-dir>
```

交叉编译

为不同于开发计算机的平台生成项目称为交叉编译。例如，在 Windows 计算机（主机平台）上为安卓（目标平台）进行构建。

对于大多数平台，使用 CMake 进行交叉编译需要一个[工具链文件](#)。除了目标平台的 Qt 版本外，它还需要开发主机的 Qt 版本。例如，您需要安装适用于 Windows 的 Qt 和适用于安卓的 Qt 才能在视窗上为安卓进行交叉编译。

从目标平台的 Qt 安装中使用，为该平台交叉编译项目：qt-cmake

Topics >

这将为目标平台配置项目。工具链文件会自动传递，并且可能还会设置其他特定于平台的变量。

指定自定义工具链文件

该脚本将 Qt 内部[工具链文件](#)传递给 CMake。此工具链文件设置了几个特定于 Qt 目标平台的变量。qt-cmake

如果您使用的是尚未在计算机上构建的 Qt 安装，则需要知道目标平台的 CMake 工具链文件的位置。qt-cmake

在这种情况下，您可以指示通过设置变量来链接加载自定义工具链文件：qt-cmakeQT_CHAINLOAD_TOOLCHAIN_FILE

```
-/Qt/6.4.0/android_armv7/bin/qt-cmake -DQT_CHAINLOAD_TOOLCHAIN_FILE=<file-path> -S <source-dir> -B <build-dir>
```

这指示Qt的内部工具链文件也加载您的自定义工具链文件。

[开始使用](#)

[导入的目标](#)

©2022 Qt Ltd. 此处包含的文档贡献是其各自所有者的版权。此处提供的文档是根据自由软件基金会发布的 [GNU 自由文档许可证 1.3](#) 版的条款进行许可的。Qt及其相应的徽标是Qt有限公司在芬兰和/或全球其他国家的[商标](#)。所有其他商标均为其各自所有者的财产。



联系我们

公司

关于我们
投资者
编辑部
职业
办公地点

发牌

条款及细则
开源
常见问题

支持服务
专业服务
合作 伙伴
训练

支持中心
下载
秦特登录
联系我们
客户成功案例

社区

为Qt做贡献
论坛
维基
下载
市场

© 2022 Qt公司

[反馈](#) [登录](#)