

[Qt 6.4](#) > [qmake Manual](#) > [Overview](#)

Overview

The qmake tool provides you with a project-oriented system for managing the build process for applications, libraries, and other components. This approach gives you control over the source files used, and allows each of the steps in the process to be described concisely, typically within a single file. qmake expands the information in each project file to a Makefile that executes the necessary commands for compiling and linking.

Describing a Project

Projects are described by the contents of project (.pro) files. qmake uses the information within the files to generate Makefiles that contain all the commands that are needed to build each project. Project files typically

[Topics](#) >

Project files can contain a number of different elements, including comments, variable declarations, built-in functions, and some simple control structures. In most simple projects, it is only necessary to declare the source and header files that are used to build the project with some basic configuration options. For more information about how to create a simple project file, see [Getting Started with qmake](#).

You can create more sophisticated project files for complex projects. For an overview of project files, see [Creating Project Files](#). For detailed information about the variables and functions that you can use in project files, see [Reference](#).

You can use application or library project templates to specify specialized configuration options to fine tune the build process. For more information, see [Building Common Project Types](#).

You can use the [Qt Creator new project wizard](#) to create the project file. You choose the project template, and Qt Creator creates a project file with default values that enable you to build and run the project. You can modify the project file to suit your purposes.

You can also use qmake to generate project files. For a full description of qmake command line options, see [Running qmake](#).

The basic configuration features of qmake can handle most cross-platform projects. However, it might be useful, or even necessary, to use some platform-specific variables. For more information, see [Platform Notes](#).

Building a Project

For simple projects, you only need to run qmake in the top level directory of your project to generate a Makefile. You can then run your platform's make tool to build the project according to the Makefile.

Using Third Party Libraries

The guide to [Third Party Libraries](#) shows you how to use simple third party libraries in your Qt project.

Precompiling Headers

In large projects, it is possible to take advantage of precompiled header files to speed up the build process. For more information, see [Using Precompiled Headers](#).

[< qmake Manual](#)[Getting Started with qmake >](#)

© 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



Forum

Wiki

Downloads

Marketplace

© 2022 The Qt Company

[Feedback](#) [Sign In](#)