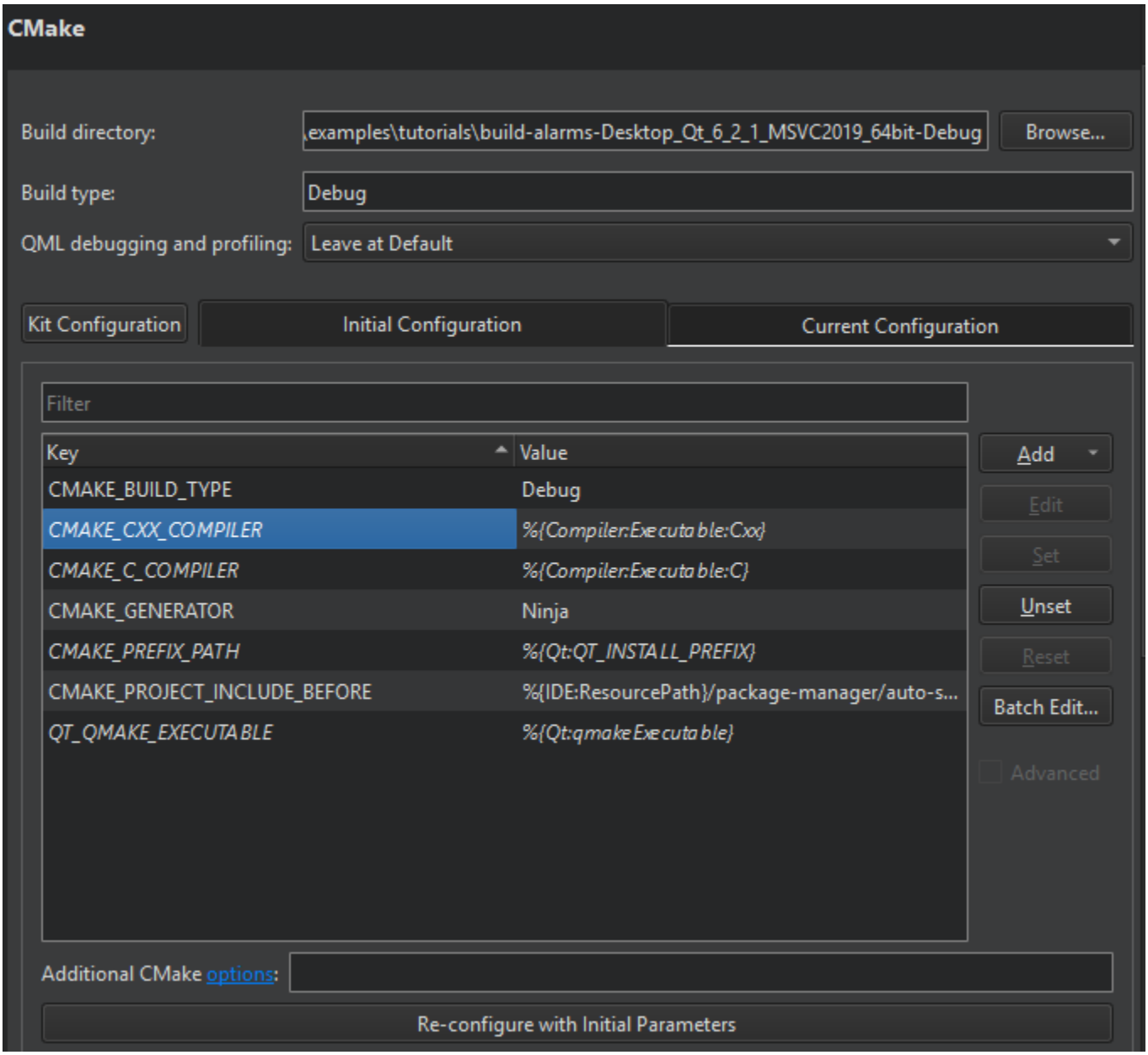
生成配置

在 Qt Creator 中配置中型到大型 CMake 项目可能是一个挑战，因为您需要传递给 CMake 才能正确配置项目的变量数量。为了简化此操作，Qt Creator 根据工具包设置为您创建初始配置，并将其显示在项目"构建**设置**"**的初始配置**中。



"**初始配置**"列出了用于首次配置 CMake 项目的变量。从工具包的 CMake 配置继承的默认值以斜体显示。变量的初始配置列表作为 *CMakeLists.txt.user* 文件保存在项目的源目录中。

您可以查看和编辑传递给 CMake 的变量的实际值。变量名称列在“**键**”列中，其当前值列在“**值**”列中。有关可用变量的更多信息，请在上下文菜单中选择“**帮助**”，或参阅 CMake： cmake 变量（7）。有关 Qt 特定变量的更多信息，请参见 CMake 变量参考。

可以在“其他 CMake 选项”中指定其他 **CMake** 选项，如 、、 或 。。有关可用选项的详细信息，请单击字段名称中的链接或参阅 CMake： cmake（1）。`--find-debug--preset--trace-expand--warn-uninitialized`

成功运行 CMake 后，可以在“当前配置”中查看和修改**当前配置**。

选择“**工具包配置**”以编辑为项目选择的生成和运行工具包的 CMake 设置。

多配置支持

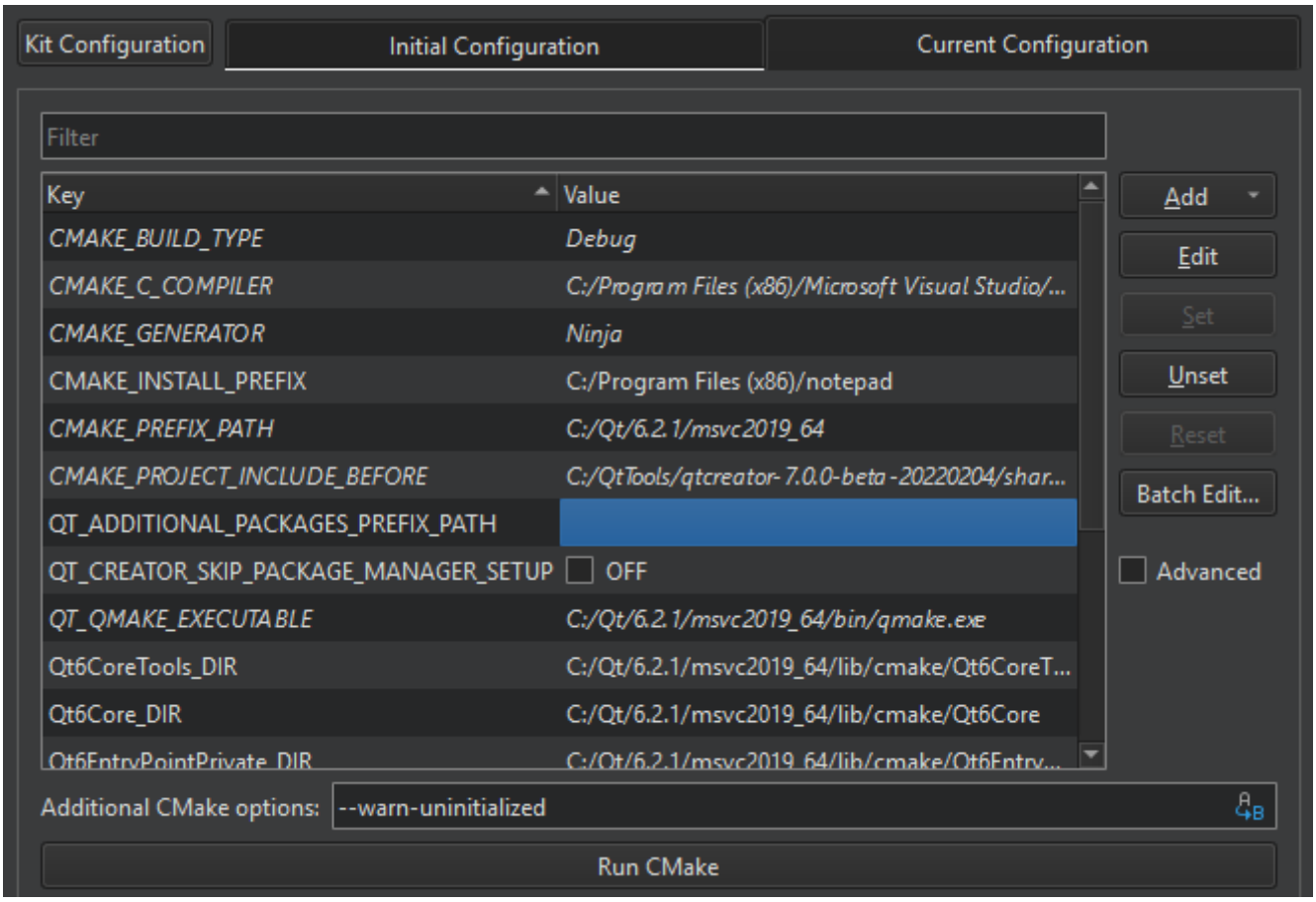
Qt Creator支持多配置生成器，如Xcode，视觉工作室和忍者多配置。这意味着您只需配置 CMake 一次，只有一个生成目录，并且可以更快地在生成类型之间切换。

但是，这意味着 Qt 创建者不能再简单地解析第一个 CMake 文件 API JSON 导出。因此，“**生成类型**”字段的值必须与单个配置生成器（Ninja、Makefile）的变量的值匹配，以确定要使用的生成器。`CMAKE_BUILD_TYPE`

使用 Qt 6 iOS 版进行开发时，仅支持 Xcode 生成器。

修改变量值

您可以查看和编辑在“**初始配置**”**或**“**当前配置**”中传递给 CMake 的变量的实际值。



To view all variables, select the **Advanced** check box.

To change the type of the selected variable, right-click the variable name in the **Key** column, and then select **Force to bool**, **Force to file**, **Force to directory**, or **Force to string** in the context menu.

To copy the name or value of the selected variable to the clipboard, select **Copy** in the context menu.

To modify the value of a variable, double-click it, or select it, and then select **Edit**. If the initial, current, and kit configuration get out of sync, select **Apply Kit Value** or **Apply Initial Configuration Value** in the context menu in **Initial Configuration** or **Current Configuration**.

You can apply actions to multiple variables at a time. To clear the selection, select **Clear Selection**.

To remove the selected variables, select **Unset**. To undo the removal, select **Set**.

To reset all the changes that you made, select **Reset**.

To modify the environment variable values for the CMake build environment, select **Batch Edit**. For more information, see Batch Editing.

To build using the current configuration, select **Run CMake**. While building, the button text changes to **Stop CMake**. Select the button to cancel the current build.

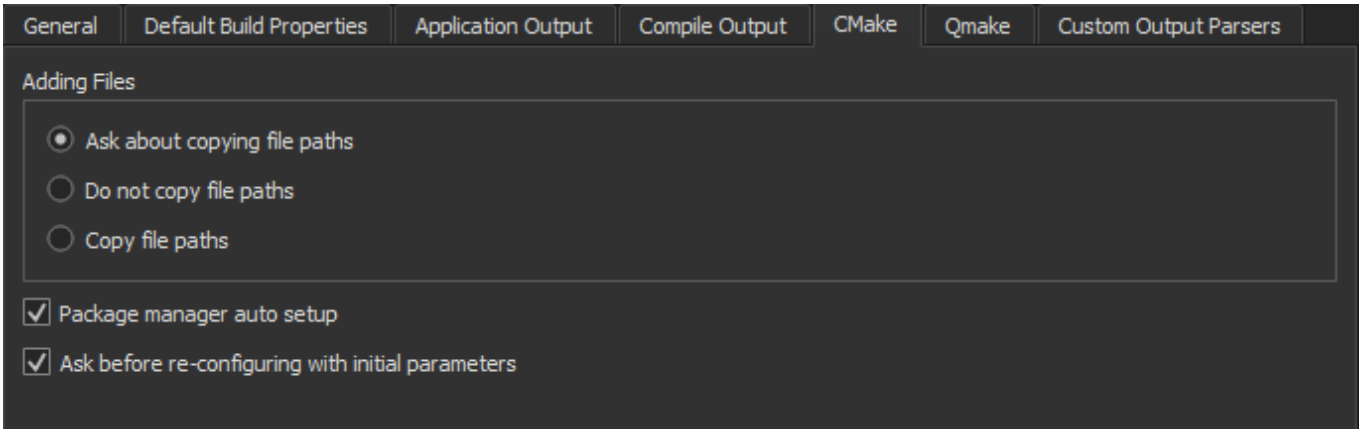
The variable values that you change are passed via to CMake, which stores the options in the CMakeCache.txt file. This means that if you remove the build directory, all the custom variables that are not part of the initial CMake configuration are also removed.`-D<option>=<value>`

To reconfigure a project using the modified variable values, select **Build** > **Clear CMake Configuration**, which removes the CMakeCache.txt file. This enables you to do a full rebuild.

Re-configuring with Initial Variables

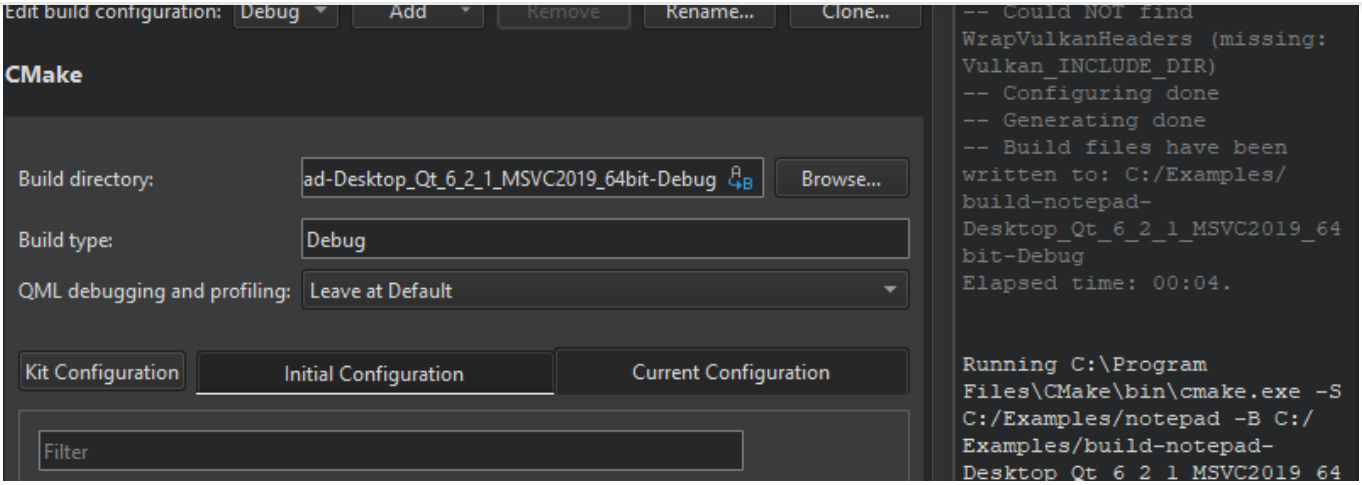
To reset CMake variables to the initial ones, select **Re-configure with Initial Variables** in **Initial Configuration**. Qt Creator deletes the current CMake configuration and runs CMake. The initial configuration values are stored in the CMakeLists.txt.user file, so deleting a build directory does not delete the initial configuration.

To be asked before Qt Creator resets the changes, select **Edit** > **Preferences** > **Build & Run** > **CMake** > **Ask before re-configuring with initial parameters**.



Viewing CMake Output

Output from CMake is displayed next to the **Build Settings** and **Run Settings** panes in the **Projects** mode.



To clear the search results, select the (**Clear**) button.

You can enter a string in the **Filter** field to filter output. To specify filtering options, select the button. You can filter output by using regular expressions or case-sensitivity. Select **Show Non-matching Lines** to hide the lines that match the filter.

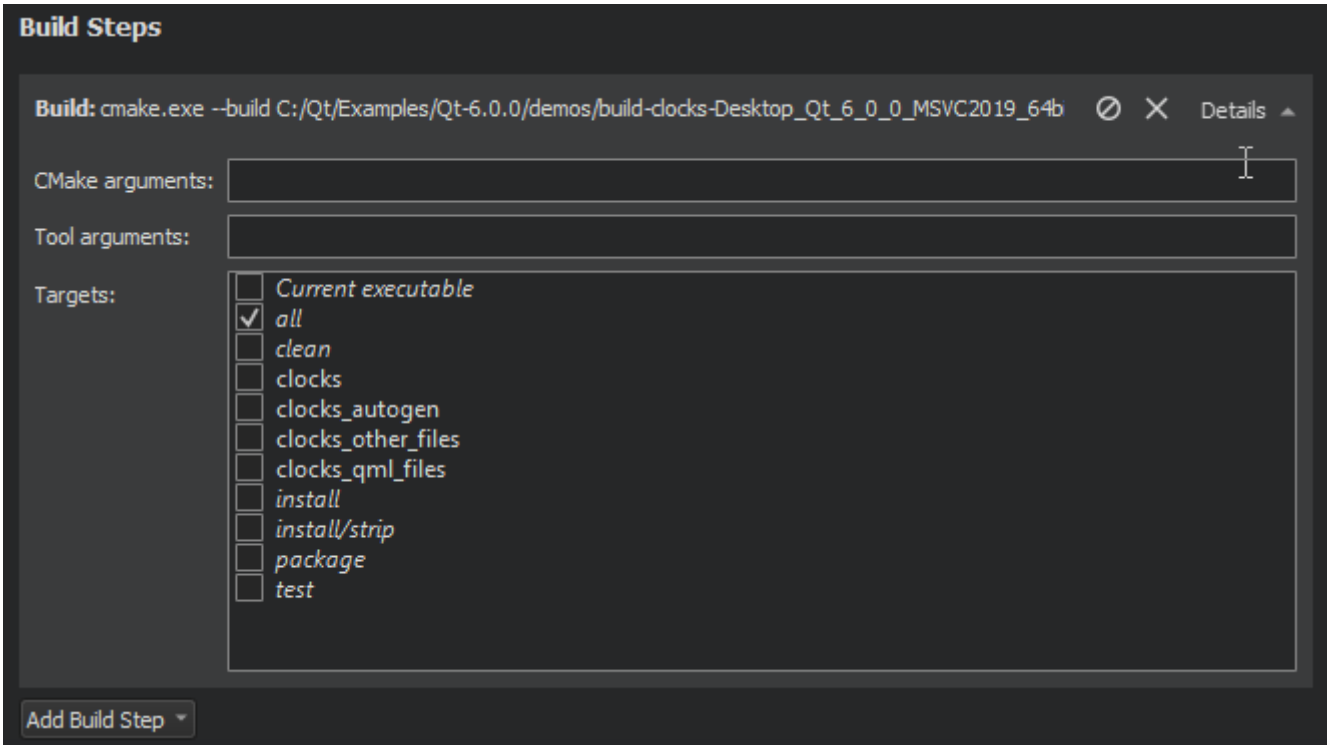
Press **Ctrl+F** to search for a string from the output.

To increase or decrease the output text size, select (**Zoom In**) or (**Zoom Out**), or press **Ctrl++** or **Ctrl+-**.

CMake Build Steps

Qt Creator builds CMake projects by running , which then runs the CMake generator specified in the project configuration: , , , or , for example. The CMake generator produces project files for Qt Creator. Multi-config generators are also supported.`cmake . --buildmakemingw32-makenmakeninja`

You can add arguments to pass to CMake and the generator and targets for the build command in **Build Steps**.

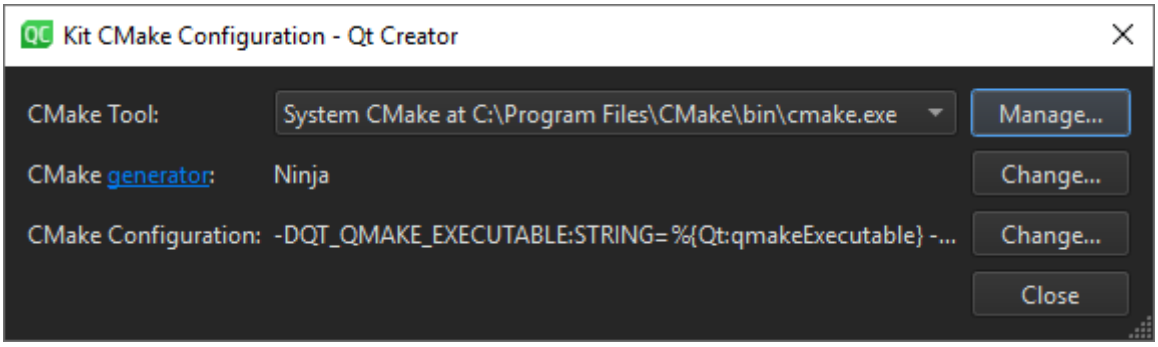


yourself.

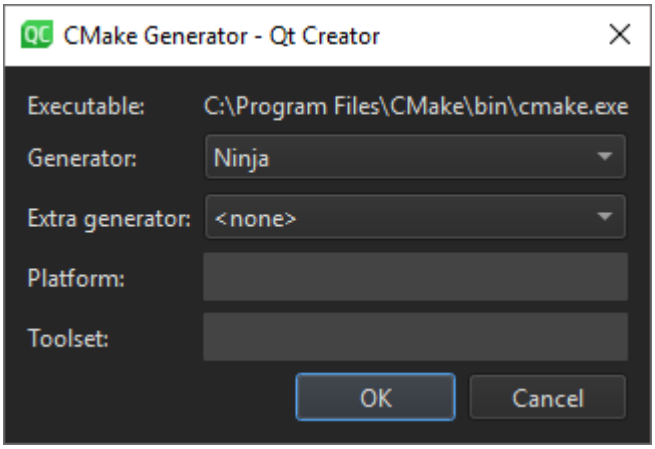
Using Ninja as a CMake Generator

To use Ninja with CMake, you must install it and select it as the CMake generator in the build and run kit:

1. Install Ninja.
2. Add the path to the Ninja executable to the value of the PATH system variable.
3. In **Projects** > **Build & Run** > **Build** > **Build Settings**, select **Kit Configuration**.



4. Select **Change** next to the **CMake generator** field to open the **CMake Generator** dialog.



5. In **Generator**, select **Ninja**.
6. Select **OK** to save your changes and close the dialog.
7. Select **Close** to close the **Kit CMake Configuration** dialog and return to **Build Settings**.

Note: To make sure that old build artifacts don't get in the way the first time you build the project after the change, select **Build** > **Rebuild Project**. This cleans up the build directory and performs a new build.

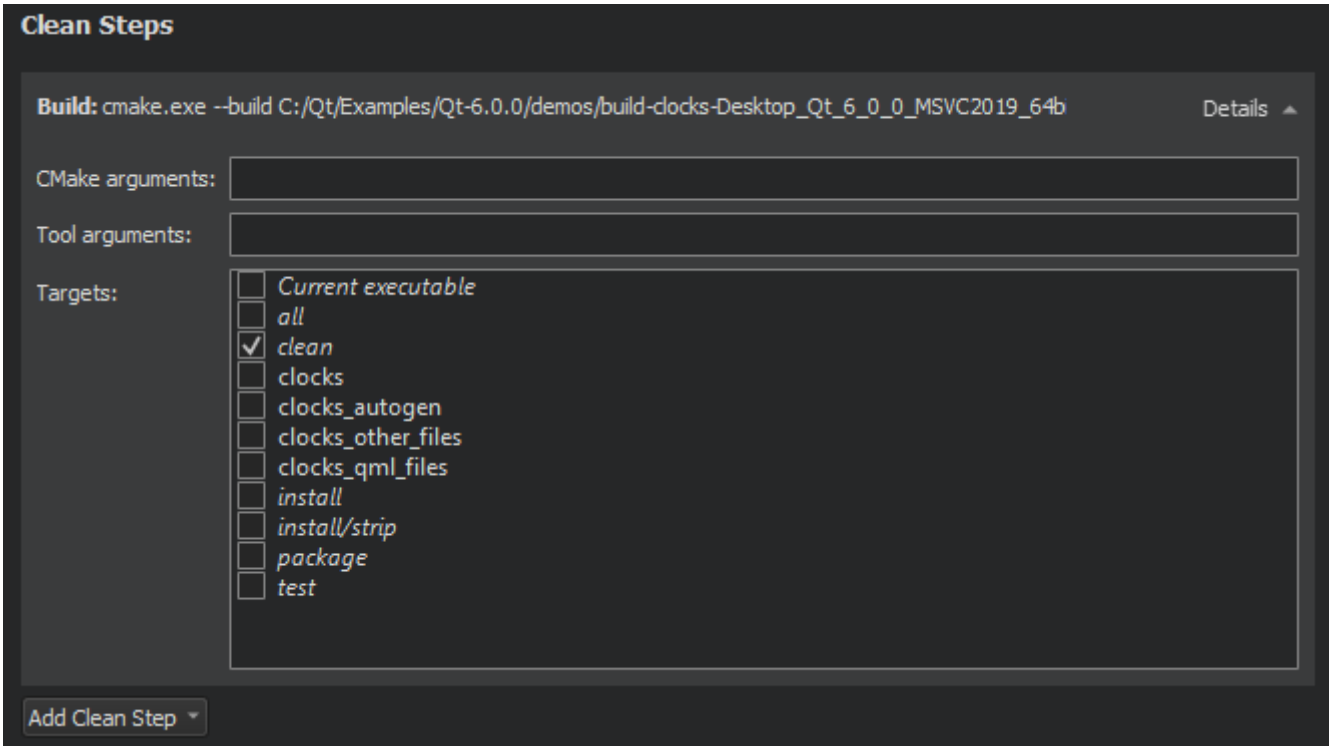
Using CMake with Conan

Qt Creator can automatically set up the Conan package manager for use with CMake.

Select **Edit** > **Preferences** > **Build & Run** > **CMake** > **Package manager auto setup** to set the value of the variable to the path to a CMake script that installs dependencies from a , , or file in the project source directory.`CMAKE_PROJECT_INCLUDE_BEFORE``conanfile.txt`conanfile.pyvcpkg.json

CMake Clean Steps

When building with CMake, you can add arguments to pass to CMake and the generator and targets for the clean command in **Clean Steps**.



The build errors and warnings are parsed and displayed in Issues.

‹ Specifying Build Settings

qmake Build Configuration ›