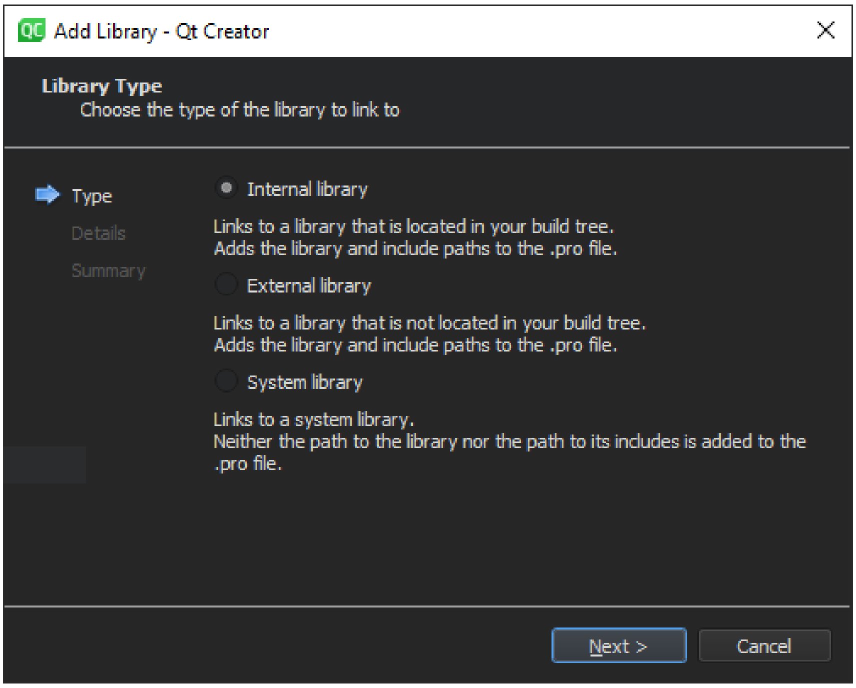
向项目添加库

除了 Qt 库之外，您还可以向项目中添加其他库。库的添加方式取决于库的类型和位置。您可以添加系统库、您自己的库或第三方库。该库可以位于当前项目的生成树中，也可以位于另一个生成树中。



Add Library - Qt Creator

Library Type
Choose the type of the library to link to

● Internal library

Links to a library that is located in your build tree.
Adds the library and include paths to the .pro file.

○ External library

Links to a library that is not located in your build tree.
Adds the library and include paths to the .pro file.

○ System library

Links to a system library.
Neither the path to the library nor the path to its includes is added to the .pro file.

由于系统库通常不会更改，并且默认情况下经常可以找到，因此在添加库或其包含时，不需要指定库或其包含的路径。可用于在编译期间查询系统库。`pkg-config`

对于您自己的库和第三方库，您需要指定路径。Qt Creator尝试猜测外部库的包含路径，但您需要检查它并在必要时进行修改。Qt 创建器会自动添加内部库的包含路径。

对于所有库，选择应用程序、库或插件的目标平台。

指定库是静态链接还是动态链接。对于静态链接的内部库，Qt Creator 会在项目文件中添加依赖项（使用 CMake 时 target_link_libraries，或使用 qmake 时PRE_TARGETDEPS）。

根据开发平台的不同，可能会自动检测到某些选项。例如，在 macOS 上，会自动检测资源库类型（**"资源库"或"框架"**），并且该选项处于隐藏状态。但是，如果您在 macOS 以外的其他平台上进行开发，并且想要针对 macOS 构建项目，则必须指定资源库类型。

Windows 上的默认约定是库的调试版本和发布版本具有相同的名称，但放置在不同的子目录中，通常称为*调试*和*发布*。如果库路径不包含这些文件夹中的任何一个，则无法选择将库放在单独文件夹中的选项。

或者，可以将字母 *d* 添加到调试版本的库名称中。例如，如果发布版本称为示例 .lib，则调试版本称为示例.lib。您可以指定为调试版本添加字母，并为发布版本删除该字母。如果库名称以 *d* 结尾，请取消选择"**删除发行版的**"**d**"**后缀**选项。

Qt Creator 支持代码完成和语法突出显示，一旦您的项目成功构建并链接到它们，添加的库。

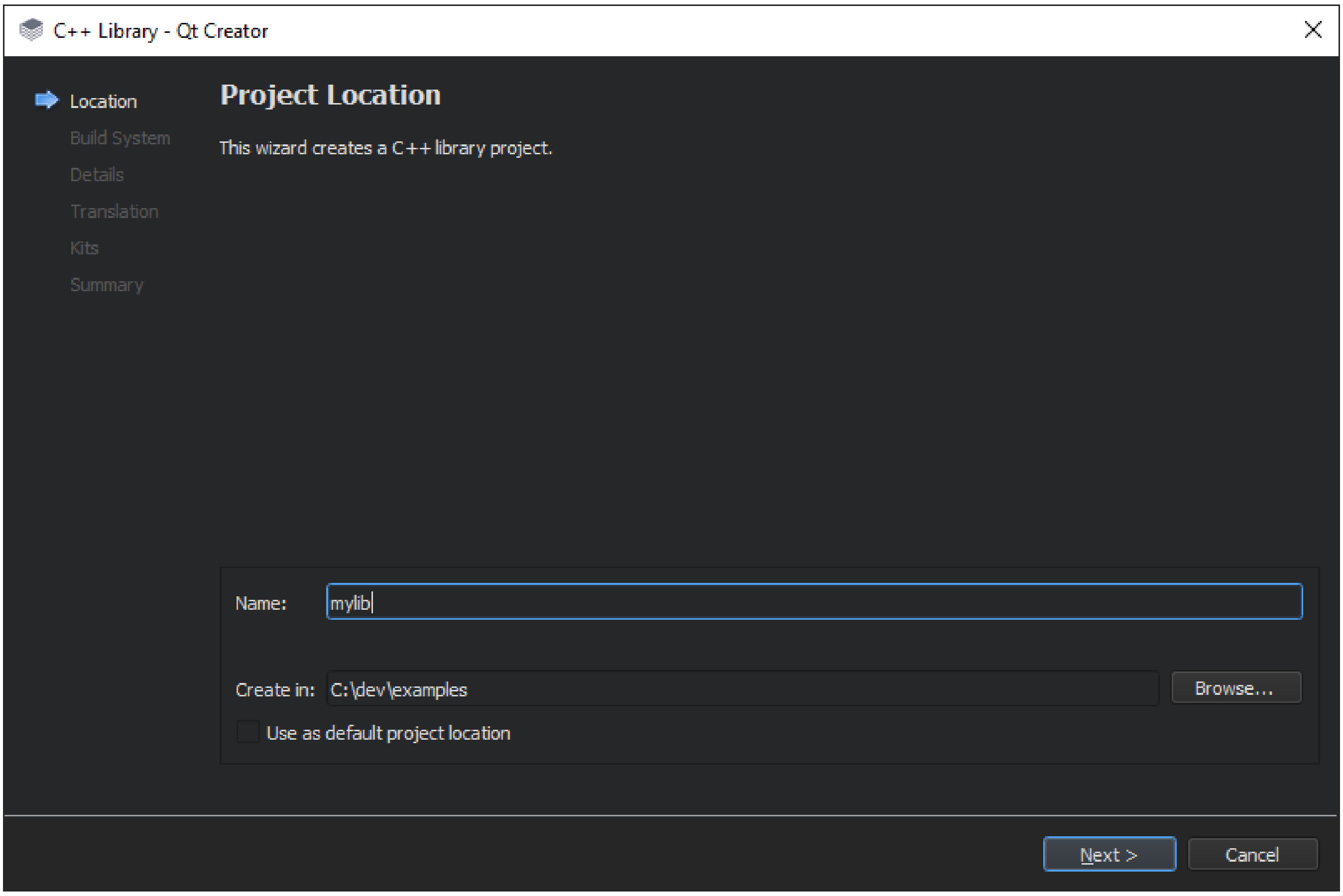
添加库

1. 在"**项目**"视图中，右键单击项目名称以打开上下文菜单，然后选择"**添加库**"。
2. 按照向导的说明进行操作。

添加内部库的示例

向项目中添加内部资源库：

1. 选择"**文件**>"**新建项目**>**库**">"**C++库**"。
2. 选择"**选择**"以打开"**项目位置**"对话框。



C++ Library - Qt Creator

Project Location

This wizard creates a C++ library project.

Name: mylib

Create in: C:\dev\examples

□ Use as default project location

3. In the **Name** field, give a name for the library. For example, **mylib**.
4. Follow the instructions of the wizard until you get to the **Project Management** dialog. In the **Add as a subproject to project** list, select a project. For example, **myapp**.
5. In the **Projects** view, right-click the project name to open the context menu and select **Add Library** > **Internal Library** > **Next**.
6. In the **Library** field, select **mylib**, and then select **Next**.
7. Select **Finish** to add the library declaration to the project file.

When using CMake, the command is added to the CMakeLists.txt file:`target_link_libraries`

```
target_link_libraries(myapp PRIVATE mylib)
```

When using qmake, the following library declaration is added to the .pro file:

```
win32:CONFIG(release, debug|release): LIBS += -L$$OUT_PWD/../../../projects/mylib/release/ -lmylib
else:win32:CONFIG(debug, debug|release): LIBS += -L$$OUT_PWD/../../../projects/mylib/debug/ -lmylib
else:unix: LIBS += -L$$OUT_PWD/../../../projects/mylib/ -lmylib
INCLUDEPATH += $$PWD/../../../projects/mylib
DEPENDPATH += $$PWD/../../../projects/mylib
win32:CONFIG(release, debug|release): PRE_TARGETDEPS += $$OUT_PWD/../../../projects/mylib/release/mylib.lib
else:win32:CONFIG(debug, debug|release): PRE_TARGETDEPS += $$OUT_PWD/../../../projects/mylib/debug/mylib.lib
else:unix: PRE_TARGETDEPS += $$OUT_PWD/../../../projects/mylib/libmylib.a
```

‹ Opening Projects Adding New Custom Wizards ›

© 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.

© 2022 The Qt Company