编写文档

当您向Qt Creator添加插件或贡献新功能时，您可能希望其他人了解它们并能够使用它们。因此，您还应该为它们提供文档。请遵循本节中的指南，以确保您的文档与Qt Creator文档的其余部分完美契合。

当你贡献一个插件时，你应该为使用Qt Creator的开发人员和开发它的开发人员编写文档。

编写以下用户文档以添加到Qt Creator手册中，或者如果您的插件位于自己的存储库中，则作为单独的插件手册：

- 概述主题，从Qt Creator用户的角度描述插件的用途
- 过程主题，描述如何将您的插件用作Qt创建器的一部分
- 参考主题，其中包含开发人员偶尔需要查找的信息（可选）

编写以下开发人员文档以添加到扩展Qt创建者手册中：

- 概述主题，从Qt Creator开发人员的角度描述插件的架构和用例
- API 文档，从代码注释生成

配置文档项目

Qt 创建者文档是使用 QDoc 编写的。有关使用 QDoc 的详细信息，请参阅 QDoc 手册。

当您将新主题作为文件放置在文档源文件夹中时，QDoc 会自动查找这些主题。但是，要使读者可以访问这些主题，还必须将它们添加到目录中，并修复从其他主题指向它们的下一页和上一页链接。`.qdoc`

创建文件夹和文件

存储库包含用于构建以下文档的源：`qtcreator`

- Qt 创建者手册
- 扩展Qt创建者手册
- Qt设计工作室手册

每个项目的源代码都存储在 Qt Creator 项目文件夹的以下子文件夹中：

- `\doc\qtcreator\src`
- `\doc\qtcreatordev\src`
- `\doc\qtdesignstudio\src`

Qt设计工作室手册基于Qt创作者手册，并附加主题。

扩展Qt创建者手册有自己的来源。此外，它还从 Qt 创建者源文件中提取 API 参考文档。将代码文档直接添加到代码源文件中。但是，您也可以将 API 概述编写为单独的文件。`.qdoc`

在相应的文件夹中为文档创建一个子文件夹。为每个主题创建一个单独的文件。`src`

最简单的方法可能是复制现有文件，将其另存为新文件，然后对其进行修改。这样，您已经拥有了必要零碎的样本，例如主题开始和结束命令、版权声明、指向下一个和上一个主题的链接以及主题标题。

将主题集成到文档中

您必须通过将新主题的链接添加到目录中和其他相关主题，将新主题集成到手册中。

要链接到该主题，您可以使用主题标题。例如：

```
\l{Integrating Topics to Documentation}
```

如果主题标题不是唯一的，则此方法不起作用。此外，如果更改标题，链接将断开。您可以通过将命令添加到主题，然后链接到目标来避免此风险。\target

显示和隐藏信息

Qt设计工作室只使用Qt创建者插件的一个子集，它有自己的特殊插件。这意味着他们的手册具有一些不同的结构。这反过来又会破坏指向上一页和下一页的导航链接。

这也意味着一些Qt创作者手册的源文件根本不需要，有些包含不适用于Qt设计工作室手册的信息。如果 QDoc 解析了所有 Qt 创作者手册源代码，它将为每个主题生成 HTML 文件，并在 Qt 设计工作室帮助编译文件中包含这些文件和它们引用的所有图像。这将不必要地增加 Qt 设计工作室帮助数据库的大小，并通过引用 Qt 设计工作室手册目录中未实际列出的文件来污染帮助索引。为避免这种情况，某些文件已从 Qt 设计工作室手册构建中排除。

从Qt设计工作室手册构建中排除源文件

要从 Qt 设计工作室手册版本中排除的目录作为 选项的值列在 中。excludedirs\doc\qtdesignstudio\config\qtdesignstudio.qdocconf

You only need to edit the values of the option if you want to show or hide all the contents of a directory. For example, if you add support for a Qt Creator plugin that was previously not supported by Qt Design Studio, you should remove the directory that contains the documentation for the plugin from the values.

To hide or show individual topics within individual files, you need to move the files in the Qt Creator Manual source () to or from the excluded directories..qdoc\doc\qtcreator\src

For example, if support for iOS were added, you would need to check whether the information about iOS support is applicable to Qt Design Studio Manual. If yes, you would need to remove the following line from the value:excludedirs

```
../../src/ios \
```

You would then use defines to hide any Qt Creator specific information from the source file in the directory.

If a directory contains some files that are needed in both manuals and some that are only needed in the Qt Creator Manual, the latter are located in a subdirectory called , which is excluded from the Qt Design Studio Manual builds.creator-only

Hiding Text in Qt Creator Manual Sources

The define is specified as a value of the option in the Qt Creator QDoc configuration file, . It is mostly used in the Qt Creator Manual sources to hide Qt Creator specific information when the Qt Design Studio Manual is built.qtcreatordefines\doc\qtcreator\config\qtcreator-project.qdocconf

The command is sometimes used to replace some Qt Creator specific text with text that applies to Qt Design Studio. For example, the following statement is needed in the Qt Creator Manual sources because the project wizards in Qt Design Studio are different from those in Qt Creator, and are therefore described in a new topic that is located in the Qt Design Studio Manual sources:\elseif-else

```
For more information, see
\if defined(qtcreator)
\l{Creating Qt Quick Projects}.
\else
\l{Creating UI Prototype Projects}.
\endif
```

Note: Section titles in the two manuals can be identical only if the page is excluded from the Qt Design Studio Manual. In this case, QDoc can correctly determine the link target. If you add a link to a section title that appears twice in the doc source files, QDoc uses the first reference to that title in the file..index

Writing About Qt Design Studio Specific Features

Qt Design Studio specific plugins and features are described in a set of doc source files located in the directory. Some files are used to include subsections in topics in the Qt Creator Manual sources.\doc\qtdesignstudio\src

Screenshots and other illustrations are stored in the directory.\qtdesignstudio\images

If you add new topics to the Qt Design Studio Manual, add links to them to the table of contents in and check the values of the navigation links around them.qtdesignstudio-toc.qdoc

Including Sections in Qt Creator Manual Sources

Qt Quick Designer is an integral part of both Qt Creator and Qt Design Studio. Therefore, most topics that describe it are needed in the manuals of both tools. You can use the command in the Qt Creator Manual sources to include files from the Qt Design Studio Manual sources when building the Qt Design Studio Manual.\include.qdocinc

For example, the following lines in the file add information about creating and using Qt Design Studio Components to the *Creating Components* topic that is

```
\if defined(qtdesignstudio)
\include qtdesignstudio-components.qdocinc creating studio components
\include qtdesignstudio-components.qdocinc studio components
\endif
```

Similarly, you can use include files to include subsections in different main level topics in the two manuals.

Updating Next and Previous Links

When you add new topics to a document, you must also change the navigation links of the topics around them.

The navigation order of the topics in the Qt Creator Manual is specified in and that of the topics in the Qt Design Studio Manual in . If you add topics to or move them around in a TOC file, you must adjust the navigation links accordingly.`\doc\qtcreator\src\qtcreator-toc.qdoc\doc\qtdesignstudio\src\qtdesignstudio-toc.qdoc`

The define is specified as a value of the option in the Qt Design Studio Manual configuration file, . It is mostly used in the Qt Creator Manual sources to specify values for the and commands depending on whether the Qt Design Studio Manual or Qt Creator Manual is being built. For example, the following statement is needed because only the Git version control system is integrated to Qt Design Studio, and information about the other systems integrated to Qt Creator is hidden:`qtdesignstudiodefinesqtcreator\doc\qtdesignstudio\config\qtdesignstudio.qdocconf\previouspage\nextpageif-else`

```
\page creator-vcs-git.html
\if defined(qtdesignstudio)
\previouspage studio-projects.html
\nextpage studio-importing-designs.html
\else
\previouspage creator-vcs-cvs.html
\nextpage creator-vcs-mercurial.html
\endif
```

Adding Documentation for Independent Plugins

You can develop Qt Creator plugins in separate repositories. Such plugins should have their own help files (.qch) that are installed and registered only if the plugin is installed.

The easiest way to set up the documentation project for an independent plugin is to copy it from an existing repository, such as the Fossil plugin, and then make the necessary changes.

Use the following naming scheme for Qt Creator plugin manuals: *Qt Creator <Plugin Name> Plugin Manual*. For example, *Qt Creator Fossil Plugin Manual*.

Writing Text

Follow the guidelines for writing Qt documentation.

The documentation must be grammatically correct English and use the standard form of written language. Do not use dialect or slang words. Use idiomatic language, that is, expressions that are characteristic for English. If possible, ask a native English speaker for a review.

Capitalizing Headings

Use the book title capitalization style for all titles and section headings (, , , and so on). For more information, see Using Book Style Capitalization.`\title\section1\section2`

Using Images

You can illustrate your documentation by using screen shots, diagrams, and other images.

Use the and QDoc commands to refer to images from the text. You do not need to add paths to image names. For example:`\image\inlineimage`

```
\image riot.png
```

Taking Screen Shots

Qt Creator has the native look and feel on Windows, Linux, and macOS, and therefore, screen shots can end up looking very different, depending on who takes

Note: Do not rely on screen shots present reasonable example values to users, but always place example values also in the text.

- Use the screen resolution of 1920x1080 (available on the most commonly used screens, as of this writing).

  Note: Use display scaling 100%.

- Use your favorite tool to take the screen shot.
- Include only the part of the screen that you need (you can crop the image also in the screen capture tool). In Qt Design Studio, close all unnecessary views to avoid clutter.
- Do not scale or resize the images in the tool because it causes reduced visual quality and bigger file size. Also, the CSS we use online scales down images if needed (their width is larger than 800 pixels).
- To highlight parts of the screen shot, use the images of numbers that are stored in in the Qt Creator repository.`qtcreator/doc/qtcreator/images/numbers`
- Before you submit PNG images to the repository, optimize them to save space.

Hightlighting Parts of the Screen

You can use number icons in screenshots to highlight parts of the screenshot (instead of using red arrows or borders, or something similar). You can then refer to the numbers in text. For an example, see the User Interface topic in the Qt Creator Manual.

This improves the consistency of the look and feel of Qt documentation, and eliminates the need to describe parts of the UI in the text because you can just insert the number of the element you are referring to in brackets.

You can find a set of images that show the numbers from 1 to 10 in the directory (or in the module sources in).`qtcreator/doc/qtcreator/images/numbersqtdocdoc/images/numbers`

To use the numbers, copy-paste the number images on the screenshot to the places that you want to refer to from text.

Icons

The Qt Documentation published online can be viewed in dark and light modes. To make the icons used in the docs visible in both modes, save icon files as gray-scale images with a transparent background in the following locations, depending on whether they are used in both manuals or just the Qt Design Studio Manual:

- `qtcreator/doc/qtcreator/images/icons` - used in the Qt Creator Manual
- `qtcreator/doc/qtdesignstudio/images/icons` - used only in the Qt Design Studio Manual

If you receive a large number of icons that are not visible in either light or dark mode or have a solid background, run the Python script from the directory. By default, the script recolors icons in . Use the option to specify the path to another icon source directory.`recolordocsicons.pysrc/tools/icons/qtcreator/doc/qtcreator/images/icons-docspath`

For example, if you saved the new icons in , switch to the folder and enter:`C:\iconconversionsqtcreator\src\tools\icons`

```
recolordocsicons.py -docspath C:\iconconversions
```

To run the script, you will need to install the following tools and add them to the PATH:

- Python 3.x (the script has been tested to work with 3.10)
- ImageMagick
- optipng

Saving Images

Save images in PNG or WebP format in the Qt Creator project folder in the or folder. Binary images can easily add megabytes to the Git history. To keep the history as small as possible, the Git post-commit hooks remind you to try to keep image file size below 50 kilobytes. To achieve this goal, crop images so that only relevant information is visible in them.`doc/qtcreator/imagesdoc/qtdesignstudio/images`

If your screenshot contains lots of colorful content or a photo, for example, consider saving it in WebP format for a smaller image file size.

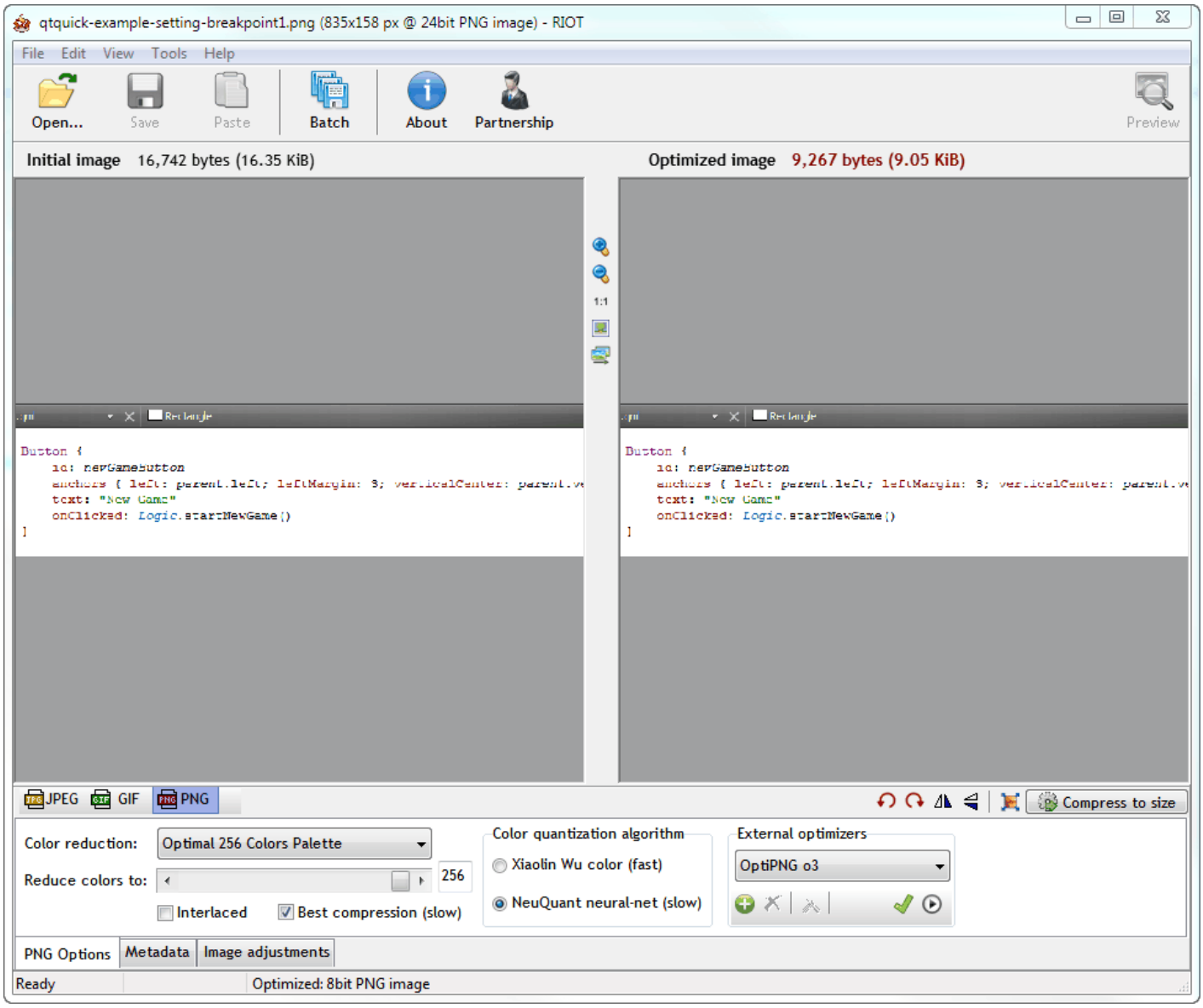
Optimizing Images

Before committing PNG images, optimize them by using an image optimization tool. Optimization should not visibly reduce image quality. If it does, try saving the image as WebP instead.

With ImageOptim, you simply drag and drop the image files to the application. The following section describes the settings to use for RIOT.

Using RIOT

Download and install RIOT.



Open your images in RIOT and use the following settings for them:

- Color reduction: Optimal 256 colors palette
- Reduce colors to: 256
- Best compression (slow)
- Color quantization algorithm: NeuQuant neural-net (slow)
- External optimizers: OptiPNG o3

Compare the initial and optimized images to check that image quality is preserved. If the image quality deteriorates, do not use color reduction (select the **True Color** option, instead).

You can also see the sizes of the initial and optimized image.

Using OptiPNG

Download and install OptiPNG.

OptiPNG is a command-line tool that you can invoke from the Qt Creator project folder (or any folder that contains your project). To optimize a screenshot, enter the following command (here, from the Qt Creator project folder):

```
optipng -o 7 -strip all doc/images/<screenshot_name>
```

Creating GIF Videos

Sometimes it is easier to explain how something works by recording a short GIF video. You can use any tool you like, for example ScreenToGif. GIF videos are typically bigger than screenshots, so try to make them as short and to the point as you can.

Use the command to add GIF files to the documentation.`\image`

Linking to Youtube Videos

You can use the macro to link to a video on Youtube. The HTML docs show a thumbnail of the video with a play button.`\youtube`

The support for the macro is defined in the and files. To use the macro, you need to save a thumbnail of the video in `.qtcreator\doc\config\macros.qdocconfqtcreator\doc\config\macros-online.qdocconfqtcreator\doc\qtcreator\images\extraimages\images`

You can use the following URL to open the thumbnail image in a browser: . The *<ID>* is the ID of the video on Youtube. For example, if the URL to the video is , the ID is . Save the image file as `.https://img.youtube.com/vi/<ID>/0.jpghttps://www.youtube.com/watch?v=9ihYeC0YJ0M&feature=youtu.be9ihYeC0YJ0M9ihYeC0YJ0M.jpg`

You must add the filename of the thumbnail file to and files in the folder.`qtcreator-extraimages.qdocconfqtdesignstudio-extraimages.qdocconf\qtcreator\doc\qtcreator\images\extraimages`

If you'll only link to the video from the Qt Creator Manual or the Qt Design Studio Manual, you'll only need to add the thumbnail filename to the file for that project.`extraimages.qdocconf`

For example, to enable linking to a video with the thumbnail filename in the Qt Design Studio Manual, the file should contain the filename:`9ihYeC0YJ0M.jpgqtdesignstudio-extraimages.qdocconf`

```
{HTML.extraimages,qhp.qtdesignstudio.extraFiles} += \
    images/commercial.png \
    images/9ihYeC0YJ0M.jpg
```

To add a link to the video in text, you would write:

```
\youtube 9ihYeC0YJ0M
```

Note: Leave out the filename extension when referring to the thumbnail.

Building Documentation

You use QDoc to build the documentation. Build the documentation before submitting any documentation patches, to check its structure, contents, and the validity of the QDoc commands. The error messages that QDoc issues are generally very useful for troubleshooting.

Setting Up Documentation Builds

You can use an installed Qt version to build the documentation. The content and formatting of documentation are separated in QDoc. The documentation configuration, style sheets, and templates have changed over time, so they differ between Qt and Qt Creator versions.

The templates to use are defined by the and configuration file. They are fetched from Qt sources by adding the following lines to the qdocconf file:`qt5/qtbase/doc/global/qt-html-templates-offline.qdocconfqt5/qtbase/doc/global/qt-html-templates-online.qdocconf`

- `include ($QT_INSTALL_DOCS/global/qt-html-templates-offline.qdocconf)` for help files
- `include ($QT_INSTALL_DOCS/global/qt-html-templates-online.qdocconf)` for publishing on the web

Note: If the styles look wrong to you when reading help files in Qt Creator or Qt Assistant, you might be using the QTextBrowser as the help engine backend instead of litehtml. For more information, see Selecting the Help Viewer Backend.

To build documentation for the sources from the master branch, use build scripts defined in the doc.pri file. You can build the docs using either the offline or online style. The offline style is used for generating HTML files included in help files (.qch), whereas the online style is used at doc.qt.io.`qtcreator`

Using CMake

When using CMake, the docs are built in the Qt Creator *build folder* or a separate doc build folder, not in the project folder.

To build docs with CMake in a separate doc build folder:

1. Create a folder for the built docs and switch to it. For example, `C:\dev\qtc-doc-build`
2. In the doc build folder, enter the following command:

```
cmake -DWITH_DOCS=ON "-DCMAKE_PREFIX_PATH=<path_to_qt>" <path_to_qtcreator_src>
```

   For example (all on one line):

```
C:\dev\qtc-doc-build>cmake -DWITH_DOCS=ON
    "-DCMAKE_PREFIX_PATH=C:\Qt\6.4.0\msvc2019_64"
    C:\dev\qtc-super\qtcreator
```

3. To also build Extending Qt Creator Manual, add the following option: `-DBUILD_DEVELOPER_DOCS=ON`
4. To also build the Qt Design Studio Manual, add the following option: `"-DCMAKE_MODULE_PATH=<absolute_path_to_qtquickdesignerrepo>/studiodata/branding/"`

   For example:

```
C:\dev\qtc-doc-build>cmake -DWITH_DOCS=ON -DBUILD_DEVELOPER_DOCS=ON
    "-DCMAKE_MODULE_PATH=C:\dev\tqtc-plugin-qtquickdesigner\studiodata\branding"
    "-DCMAKE_PREFIX_PATH=C:\Qt\6.4.0\msvc2019_64"
    C:\dev\qtc-super\qtcreator
```

5. To build the docs using the online style, use the following option instead of : `-DWITH_DOCS=ON-DWITH_ONLINE_DOCS=ON`

   For example:

```
C:\dev\qtc-doc-build>cmake -DWITH_ONLINE_DOCS=ON
    -DBUILD_DEVELOPER_DOCS=ON
    "-DCMAKE_MODULE_PATH=C:\dev\tqtc-plugin-qtquickdesigner\studiodata\branding"
    "-DCMAKE_PREFIX_PATH=C:\Qt\6.4.0\msvc2019_64"
    C:\dev\qtc-super\qtcreator
```

   Note: If you already ran CMake in a folder and want to switch to only online docs in that folder, you need to turn the offline docs off again: `-DWITH_DOCS=ON`

```
cmake -DWITH_DOCS=OFF -DWITH_ONLINE_DOCS=ON
```

6. Enter the following doc build command to build both HTML docs and the help files (.qch):

```
cmake --build . --target docs
```

7. Alternatively, to build only the HTML docs, enter:

```
cmake --build . --target html_docs
```

Note: You can enter to open the graphical CMake configuration tool, where you can select build options.`cmake-gui`

The HTML files for the documentation are generated in the following folders:

- `doc/html/qtcreator`

- doc/html/qtcreator-online
- doc/html/qtcreator-dev-online
- doc/html/qtdesignstudio-online

The help files () are generated in the folder or in the folder on macOS.`.qchshare/doc/qtcreator<application_name>.app/Contents/Resources/doc\`

You can view the HTML files in a browser and the help files in the Qt Creator **Help** mode. For more information about adding the help files to Qt Creator, see Adding External Documentation.

Additional Build Commands

Besides and , you can use the following build targets:`docshtml_docs`

- `html_docs_<doc_config_file_name>` - build the document (qtcreator/ qtcreator-dev/qtdesignstudio) in help format, but do not generate a help file (.qch)
- `html_docs_<doc_config_file_name>-online` - build the document (qtcreator/qtcreator-dev/qtdesignstudio) in online format
- `qch_docs_<doc_config_file_name>` - build the document (qtcreator/ qtcreator-dev/qtdesignstudio) in help format and generate a help file

‹ External Tool Specification Files

Qt Creator Coding Rules ›

© 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.

The Qt Company

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