添加Qt设计器插件

您可以使用 Qt API 来创建扩展 Qt 应用程序的插件。这使您能够将自己的小部件添加到Qt设计器中。将插件包含在应用程序中的最灵活的方法是将其编译为单独交付的动态库，并在运行时检测和加载。

应用程序可以检测存储在标准插件子目录中的插件。有关如何创建和定位插件以及更改默认插件路径的更多信息，请参阅如何创建Qt插件。

有关如何为 Qt 设计器创建插件的更多信息，请参阅将自定义小部件与 Qt 设计器配合使用。

查找 Qt 设计器插件

Qt Designer从标准位置获取插件，并加载与其构建密钥匹配的插件。Qt设计器既作为独立的应用程序交付，又集成到Qt创建器中。放置插件的正确文件夹取决于您使用的是独立的 Qt 设计器还是集成的 Qt 设计器。

集成的 Qt 设计器从 Windows 和 Linux 上的 Qt 创建者安装目录中的目录中获取插件。有关如何在 macOS 上配置插件的信息，请参阅在 macOS 上配置 Qt 设计器插件。`\bin\plugins\designer`

要检查哪些插件已成功加载，哪些插件失败，请选择“**工具**”>**窗体编辑器**>**关于Qt Designer插件**“。

独立的Qt设计器是用于构建项目的Qt库的一部分，位于Qt安装目录中。它从 的子目录获取插件。要检查哪些插件已成功加载，哪些插件失败，请选择**帮助**>**关于插件**。`<Qt_version>\<compiler>\bin\plugins\designerbin`

在苹果操作系统上配置 Qt 设计器插件

在 macOS 上，必须构建 GUI 应用程序并从捆绑包运行。分发包是一种目录结构，在 Finder 中查看时显示为单个实体。应用程序的捆绑包通常包含可执行文件及其所需的所有资源。

Qt创建器使用其自己的一组Qt库，这些库位于捆绑包中，因此，您需要配置要与Qt创建器一起使用的Qt设计器插件。有关如何将应用程序部署到 macOS 的更多信息，请参阅适用于 macOS 的 Qt - 部署。

以下示例说明了如何配置版本 5.2.1 的 Qwt - 用于技术应用程序的 Qt 小部件库，以便与 Qt Creator 一起使用：

1. 要检查 Qwt 库中使用的路径，请输入以下命令：`otool`

   ```
   otool -L /Developer/Applications/Qt/plugins/designer/libqwt_designer_plugin.dylib
   ```

   Qwt 5.2.1 的输出表明该插件使用 Qt 核心库（Qt设计器、QtScript、QtXml、QtGui 和 QtCore）和库特库。

   ```
   /Developer/Applications/Qt/plugins/designer/libqwt_designer_plugin.dylib:
           libqwt_designer_plugin.dylib (compatibility version 0.0.0, current version 0.0.0)
           libqwt.5.dylib (compatibility version 5.2.0, current version 5.2.1)
           QtDesigner.framework/Versions/4/QtDesigner (compatibility version 4.6.0, current version 4.6.2)
           QtScript.framework/Versions/4/QtScript (compatibility version 4.6.0, current version 4.6.2)
           QtXml.framework/Versions/4/QtXml (compatibility version 4.6.0, current version 4.6.2)
           QtGui.framework/Versions/4/QtGui (compatibility version 4.6.0, current version 4.6.2)
           QtCore.framework/Versions/4/QtCore (compatibility version 4.6.0, current version 4.6.2)
           /usr/lib/libstdc++.6.dylib (compatibility version 7.0.0, current version 7.9.0)
           /usr/lib/libgcc_s.1.dylib (compatibility version 1.0.0, current version 438.0.0)
           /usr/lib/libSystem.B.dylib (compatibility version 1.0.0, current version 125.0.1)
   ```

- `libqwt_designer_plugin.dylib`自`Qt Creator.app/Contents/PlugIns/designer`
- `libqwt.*.dylib`自`Qt Creator.app/Contents/Frameworks`

输入以下命令：

```
sudo cp /Developer/Applications/Qt/plugins/designer/libqwt_designer_plugin.dylib \
        /Developer/Applications/Qt/Qt\ Creator.app/Contents/MacOS/designer
sudo cp -R /usr/local/qwt-5.2.1/lib/*  \
        /Developer/Applications/Qt/Qt\ Creator.app/Contents/Frameworks/
```

3. 输入以下命令以检查 Qwt 库使用的库：`otool`

```
otool -L /usr/local/qwt-5.2.1/lib/libqwt.5.dylib
```

该命令返回以下输出：

```
/usr/local/qwt-5.2.1/lib/libqwt.5.dylib:
        libqwt.5.dylib (compatibility version 5.2.0, current version 5.2.1)
        QtGui.framework/Versions/4/QtGui (compatibility version 4.6.0, current version 4.6.2)
        QtCore.framework/Versions/4/QtCore (compatibility version 4.6.0, current version 4.6.2)
        /usr/lib/libstdc++.6.dylib (compatibility version 7.0.0, current version 7.9.0)
        /usr/lib/libgcc_s.1.dylib (compatibility version 1.0.0, current version 438.0.0)
        /usr/lib/libSystem.B.dylib (compatibility version 1.0.0, current version 125.0.1)
```

4. Enter the following command to fix the references of the libraries:`install_name_tool`

```
cd /Developer/Applications/Qt/Qt\ Creator.app/Contents/MacOS/designer
sudo install_name_tool -change
    QtCore.framework/Versions/4/QtCore \
    @executable_path/../Frameworks/libQtCore.4.dylib \
    libqwt_designer_plugin.dylib
sudo install_name_tool -change QtGui.framework/Versions/4/QtGui \
    @executable_path/../Frameworks/libQtGui.4.dylib \
    libqwt_designer_plugin.dylib
sudo install_name_tool -change QtXml.framework/Versions/4/QtXml \
    @executable_path/../Frameworks/libQtXml.4.dylib \
    libqwt_designer_plugin.dylib
sudo install_name_tool -change QtScript.framework/Versions/4/QtScript \
    @executable_path/../Frameworks/libQtScript.4.dylib \
    libqwt_designer_plugin.dylib
sudo install_name_tool -change QtDesigner.framework/Versions/4/QtDesigner \
    @executable_path/../Frameworks/libQtDesigner.4.dylib \
    libqwt_designer_plugin.dylib
sudo install_name_tool -change libqwt.5.dylib \
    @executable_path/../Frameworks/libqwt.5.dylib \
    libqwt_designer_plugin.dylib

cd /Developer/Applications/Qt/Qt\ Creator.app/Contents/Frameworks
sudo install_name_tool -change \
    QtCore.framework/Versions/4/QtCore \
    @executable_path/../Frameworks/libQtCore.4.dylib \
    libqwt.5.2.1.dylib
sudo install_name_tool -change \
    QtGui.framework/Versions/4/QtGui \
    @executable_path/../Frameworks/libQtGui.4.dylib \
    libqwt.5.2.1.dylib
```

Matching Build Keys

The Qt Creator that is included in pre-built Qt packages on Windows is built with the Microsoft Visual Studio compiler, whereas the version of Qt shipped for building applications is configured and built to use the MinGW/g++ compiler. Plugins built by using this version of Qt cannot be loaded by Qt Creator because the build-keys do not match. The plugins can only be used in the standalone version of Qt Designer. Choose **Help** > **About Qt Creator** to check the Qt version Qt Creator was built with.

To use Qt Designer plugins that were built for the shipped Qt version, make sure that Qt Creator is built with the same compiler by either recompiling Qt Creator using MinGW or recompiling Qt with Microsoft Visual Studio, depending on which configuration you want to use for your applications.

‹ Developing Widget Based Applications Optimizing Applications for Mobile Devices ›

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

© 2022 The Qt Company Feedback Sign In