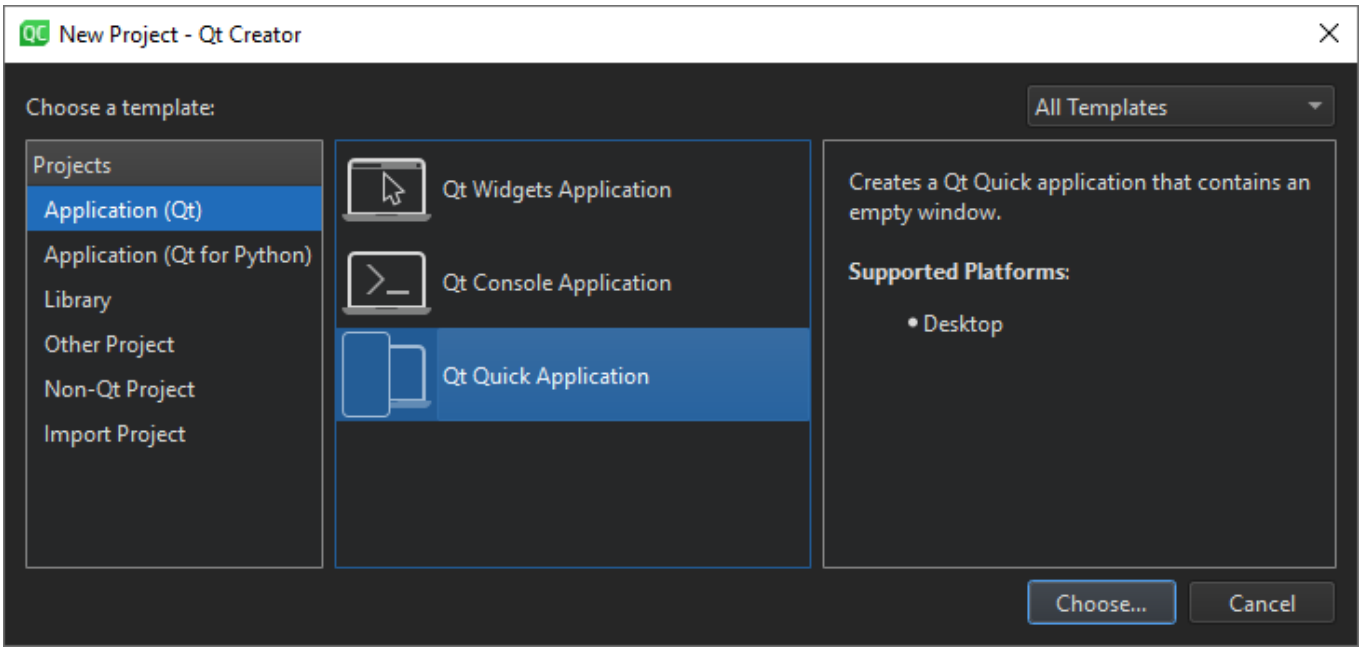
搜索

创建Qt快速项目



下表列出了用于从头开始创建新的Qt Quick项目的向导模板。

类别	向导模板	目的
应用（Qt）	Qt快速申请	创建一个可以同时包含 QML 和C++代码的 Qt Quick 2 应用程序项目。您可以生成应用程序并将其部署到桌面、嵌入式和移动目标平台。
应用程序（Qt for Python）	Qt for Python - Qt Quick Application	创建一个包含空 Qt 快速应用程序的 Python 项目。
其他项目	Qt快速UI原型	使用包含主视图的单个 QML 文件创建 Qt 快速 UI 项目。您可以在QML场景预览工具中预览Qt Quick 2 UI项目。您不需要生成它们，因为它们不包含任何C++代码。 此项目类型与Qt Design Studio兼容。但是，仅当要进行原型设计时，才使用此模板。不能使用此模板创建完整的应用程序。 Qt Quick UI项目无法部署到嵌入式或移动目标平台。对于这些平台，请改为创建Qt Quick应用程序。
图书馆	Qt快速2扩展插件	创建C++插件，使提供可以动态加载到Qt Quick 2应用程序中的扩展成为可能。

装的。

Qt Creator创建必要的样板文件。某些文件特定于特定目标平台。

创建Qt快速应用程序

1. 选择**文件**>**新建项目**>**应用程序 （Qt）**>**Qt 快速应用程序**>**选择**。
2. 在"项目位置"对话框的**"名称"**字段中，输入**项目**的名称。请记住，以后无法轻松更改项目名称。
3. 在"**创建位置**"字段中，输入项目文件的路径。选中"**用作**默认项目位置"复选框以默认在此文件夹中创建新项目。您可以稍后毫无问题地移动项目文件夹。
4. 选择**下一步**（或在 macOS 上**继续**）以打开定义**构建系统**对话框。
5. 在"生成系统"字段中，选择要用于生成和运行项目的**生成系统**：qmake、CMake 或Qbs。
6. 选择"**下一步"**以打开"**定义项目详细信息**"对话框。
7. 在"**所需的最低 Qt**版本"字段中选择要用于开发的 Qt 版本。Qt 版本确定在 QML 文件中使用的 Qt 快速导入。
8. 选中"使用 Qt 虚拟键盘"复选框以向应用程序添加对Qt 虚拟键盘的支持

Qt创建者手册8.0.2
Topics >

条错误消息。您可以使用Qt维护工具安装Qt虚拟键盘。

9. 选择"**下一步"**以打开**"翻译文件**"对话框。
10. 在"**语言**"字段中，选择您计划将应用程序翻译成的语言。您可以稍后通过编辑项目文件来添加其他语言。
11. 在 翻译文件 字段中，您可以编辑将为所选语言生成的翻译源**文件**的名称。
12. 选择"**下一步**"以打开**"套件选择**"对话框。
13. 为要为其构建应用程序的平台选择工具包。

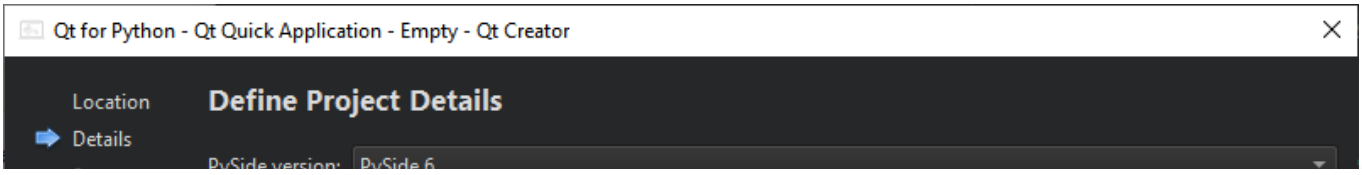
> **注意：**如果在**Edit**>Preferences>Kits（在 Windows 和 Linux 上）或**Qt Creator**>**Preferences**>**Kits**（在 macOS 上）中指定了工具包，则会列出它们。有关详细信息，请参阅添加工具包。

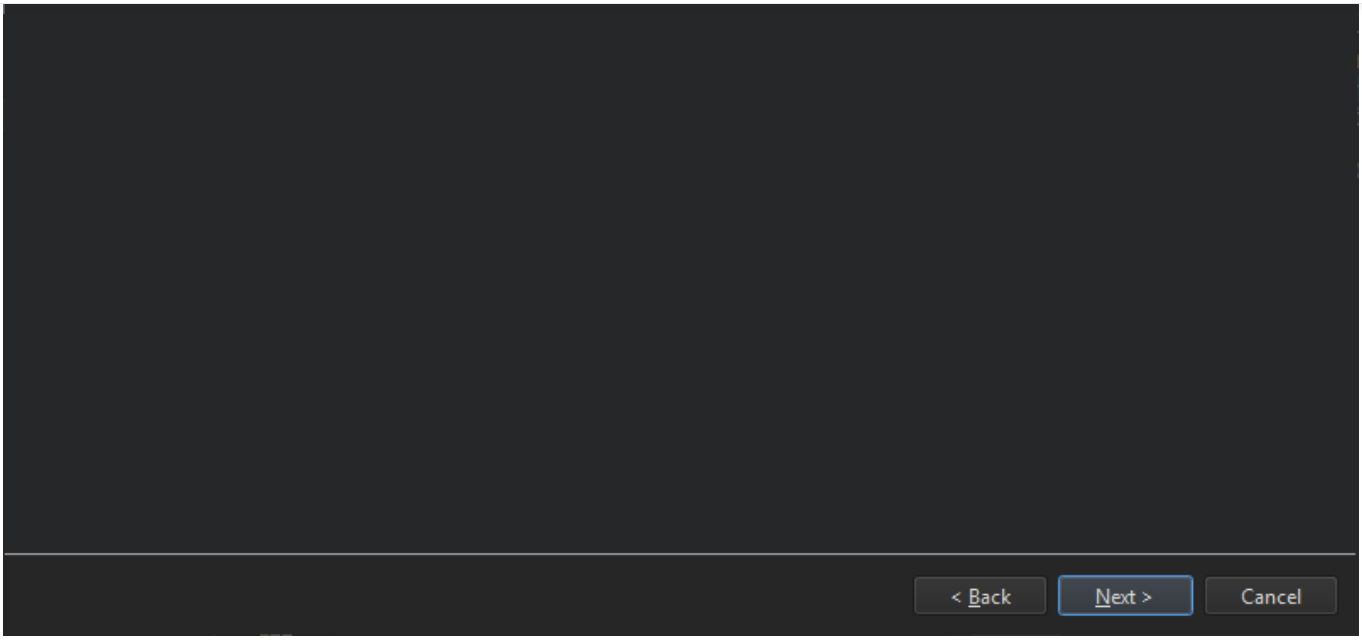
14. 选择"**下一步"**以打开"**项目管理**"对话框。
15. 查看项目设置，然后选择**"完成**"（在 Windows 和 Linux 上）或**"完成**"（在 macOS 上）以创建项目。

Qt Creator 会创建一个 QML 文件 *main.qml*，您可以在**编辑**模式下对其进行修改。

创建基于Qt快速的Python应用程序

Qt for Python - Qt Quick Application - Empty向导使您能够创建包含主 QML 文件的 Python 项目。指定运行应用程序的最低 PySide 版本。





向导将以下导入添加到源文件中，以提供对QGuiApplication和QQmlApplicationEngine 的访问：

```
import sys
from pathlib import Path

from PySide6.QtGui import QGuiApplication
from PySide6.QtQml import QQmlApplicationEngine
```

该向导还添加一个 main 函数，在该函数中，它创建一个 QGuiApplication 实例并将系统参数传递给QGuiApplication对象：

```
if __name__ == "__main__":
    app = QGuiApplication(sys.argv)
...
```

主类中的以下行创建一个QQmlApplicationEngine实例，并将生成的 QML 文件加载到引擎对象：

```
engine = QQmlApplicationEngine()
qml_file = Path(__file__).resolve().parent / "main.qml"
engine.load(qml_file)
```

最后，向导将添加检查文件是否已成功加载的代码。如果加载文件失败，应用程序将退出并显示错误代码。如果加载成功，向导将调用该方法进入Qt主循环并开始执行Qt代码：`app.exec()`

```
if not engine.rootObjects():
    sys.exit(-1)
sys.exit(app.exec())
```

在**编辑**模式下打开.qml文件以设计Qt Quick UI，或使用Qt Design Studio。

创建Qt快速UI项目

例如，Qt Quick UI 原型项目对于测试或原型用户界面非常有用，或者用于为 QML 编辑设置单独的项目。不能将它们用于应用程序开发，因为它们不包含：

- C++代码
- 资源文件 （.qrc）
- 将应用程序部署到设备所需的代码

有关如何将Qt Quick UI原型项目转换为Qt快速应用程序项目的更多信息，请参阅将UI项目转换为应用程序。

要创建Qt Quick UI原型项目，请执行以下操作：

1. 选择**"文件>新建项目>其他项目>Qt 快速 UI 原型"**。
2. 选择"选择"以打开"**项目位置"**对话框。
3. 在"**名称**"字段中，输入应用程序的名称。
4. 在"**创建位置**"字段中，输入项目文件的路径。选中"**用作**默认项目位置"复选框以默认在此文件夹中创建新项目。
5. 选择**"下一步"**（或在 macOS 上**选择"继续"**）以打开"**定义项目详细信息**"对话框。
6. 在"**所需的最低 Qt**版本"字段中，选择要用于开发的 Qt 版本。Qt 版本确定在 QML 文件中使用的 Qt 快速导入。

   您可以稍后添加导入，将 Qt Quick 基本类型与 Qt 快速控件、Qt 快速对话框和 Qt 快速布局（从 Qt 5.1 开始可用）相结合。
7. 选中"使用 Qt 虚拟键盘"复选框以向应用程序添加对Qt 虚拟键盘的支持。

   > **注意：**如果在安装 Qt 时尚未安装 Qt 虚拟键盘模块，则当您尝试打开*main.qml*时会出现一条错误消息。
8. 选择"**下一步**"以打开**"套件选择"**对话框。
9. 为要为其构建应用程序的平台选择工具包。

   > **注意：**如果在**Edit**>Preferences>Kits（在 Windows 和 Linux 上）或**Qt Creator**>**Preferences**>**Kits**（在 macOS 上）中指定了工具包，则会列出它们。有关详细信息，请参阅添加工具包。
10. 选择"**下一步"**以打开"**项目管理**"对话框。
11. 查看项目设置，然后选择**"完成"**（在 Windows 和 Linux 上）或**"完成"**（在 macOS 上）以创建项目。

Qt Creator创建以下文件：

- .qmlproject 项目文件定义了项目文件夹中的所有 QML、JavaScript 和图像文件都属于该项目。因此，您不需要单独列出项目中的所有文件。
- .qml 文件定义 UI 项，例如组件或整个应用程序 UI。
- ui.qml 文件定义应用程序 UI 的表单。如果选中了"**使用 .ui.qml 文件**"复选框，则会创建此文件。

‹ 开发Qt快速应用程序 使用Qt快速设计器 ›