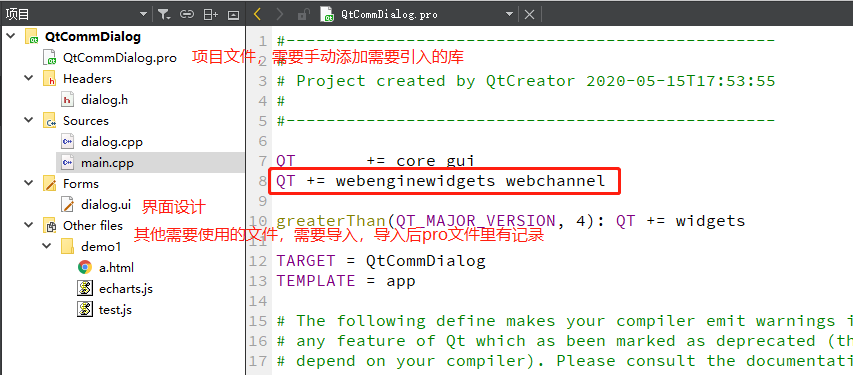
Qt 文档

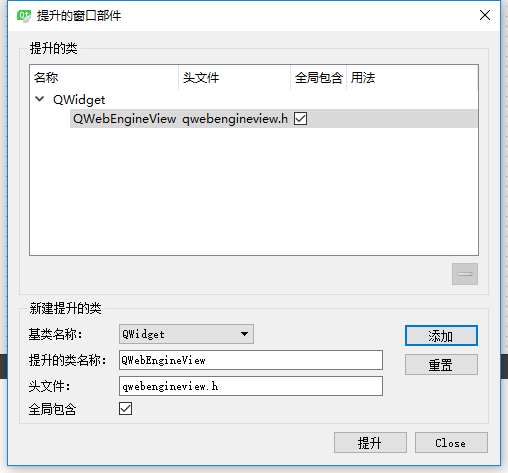
1. 下载、安装、配置
   1. <http://download.qt.io/archive/qt/5.9/5.9.0/> 官方网站下载合适的版本；

本次下载 [qt-opensource-windows-x86-5.9.0.exe](http://download.qt.io/archive/qt/5.9/5.9.0/qt-opensource-windows-x86-5.9.0.exe)；并安装；

* 1. 编译器下载，本次要用到MSVC的编译器，下载安装visual studio 2017 版本（某博客说该版本支持 QWebEngineView）
  2. QT 安装过程中缺少组件可在QT安装目录下找到 MaintenanceTool.exe 添加；
     1. 需要注册，或者更新 MaintenanceTool.exe;
     2. 需要添加临时镜像 <http://mirrors.ustc.edu.cn/qtproject/online/qtsdkrepository/windows_x86/root/qt/>
     3. 组件选择时勾选要使用的QT相关的组件尤其是 webengine；
  3. 环境变量配置；将QT编译器的include 和 bin 添加到系统 path;(有说法不用配置)
     1. C:\Qt\Qt5.9.0\5.9\msvc2017\_64\include
     2. C:\Qt\Qt5.9.0\5.9\msvc2017\_64\bin
  4. 打开Qt Creator ->工具->选项->构建和运行 配置Kit;
     1. 主要包括 编译器、调试器、Qt 版本 三个信息；
     2. 
  5. 打开QT安装目录下的Examples ;
     1. C:\Qt\Qt5.9.0\Examples\Qt-5.9\webenginewidgets
     2. 编译运行其中的 minimal项目，成功即可。

1. QT 项目结构简介



1. QT 使用QWebEngineView 显示 html
   1. 工程文件 .pro 中添加 QT += webenginewidgets webchannel
      1. Webenginewidgets （web显示类，用于显示web界面)
      2. Webchannel（web数据通道类，用于数据通信）
   2. 双击打开界面ui，在UI编辑器中添加一个widget,（建议改名，ganttChart）并将widget提升为“QWebEngineView” ；
   3. 在工程中引入html文件和相关的js等资源；项目右键->Add Existing Directory
   4. 在代码中载入html; (添加对应的头文件)

QString htmlpath ="D:/Qt\_project/QtCommDialog/demo1/a.html";

ui->ganttChart->load(QUrl(htmlPath));

* 1. 选择MSVC 2017 编译器运行即可显示html页面；

1. QT 调用前端 js
   1. QT是通过QWebEngineView来调用HTML页面的，而HTML是通过JS来调用ECharts。因此QT可以通过QWebEngineView调用加载的html页面中的JS方法，先将HTML页面中JS设置ECharts属性的那段代码封装成一个方法供QWebEngineView调用。
   2. 在QT工程中的UI界面中增加一个QPushButton并添加信号和槽函数，在槽函数中添加如下代码；

槽函数代码： json 数据；

QJsonObject seriesData;

QJsonArray data1 = {1, 3, 9, 27, 81, 247};

seriesData.insert("data1", data1);

QString optionStr = QJsonDocument(seriesData).toJson();

QString js = QString("init2(%1)").arg(optionStr);

ui->ganttChart->page()->runJavaScript(js);

js 代码：

function init2(str){

// 指定图表的配置项和数据

var option = {

title: {

text: '第一个 ECharts 实例'

},

tooltip: {},

legend: {

data:['销量']

},

xAxis: {

data: ["衬衫","羊毛衫","雪纺衫","裤子","高跟鞋","袜子"]

},

yAxis: {},

series: [{

name: '销量',

type: 'bar',

data: str["data1"]

}]

};

// 使用刚指定的配置项和数据显示图表。

myChart.setOption(option);

}

1. Qt与前端交互
2. 信号槽函数使用

signals:

void findNext(const QString &str, Qt::CaseSensitivity cs);

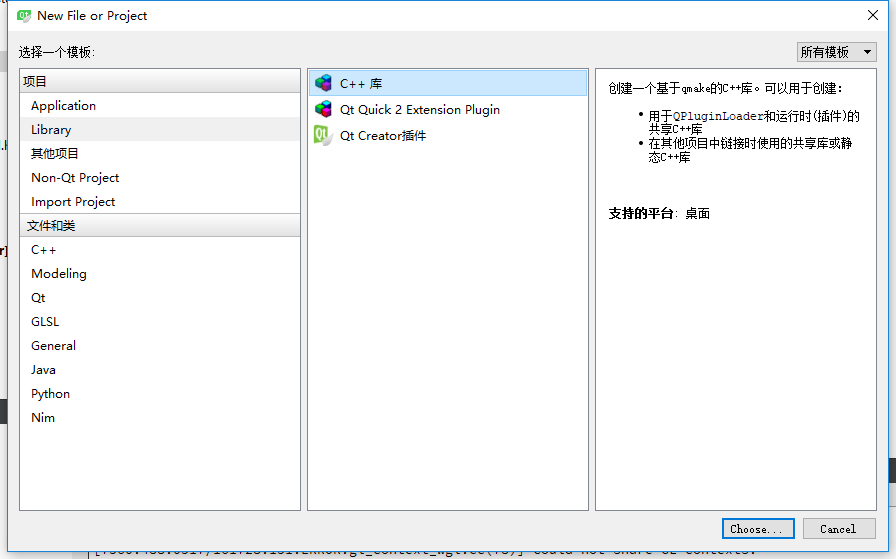
void findPrevious(const QString &str, Qt::CaseSensitivity cs);

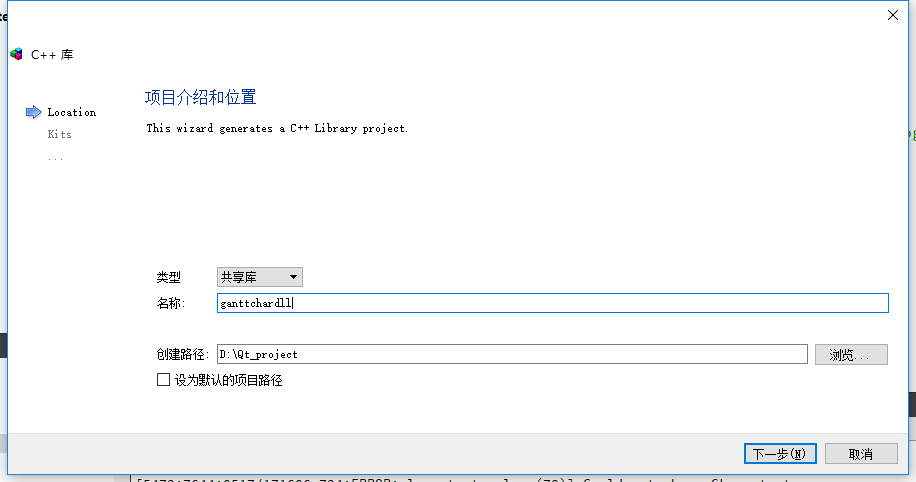
private slots:

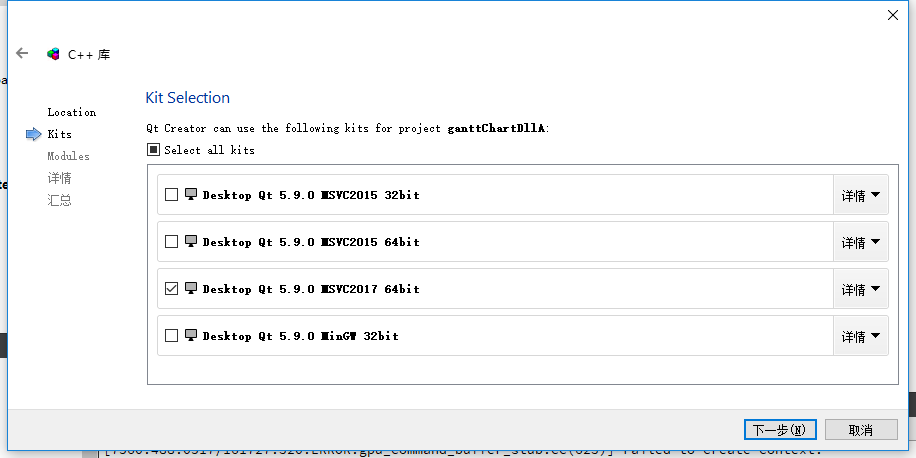
void findClicked();

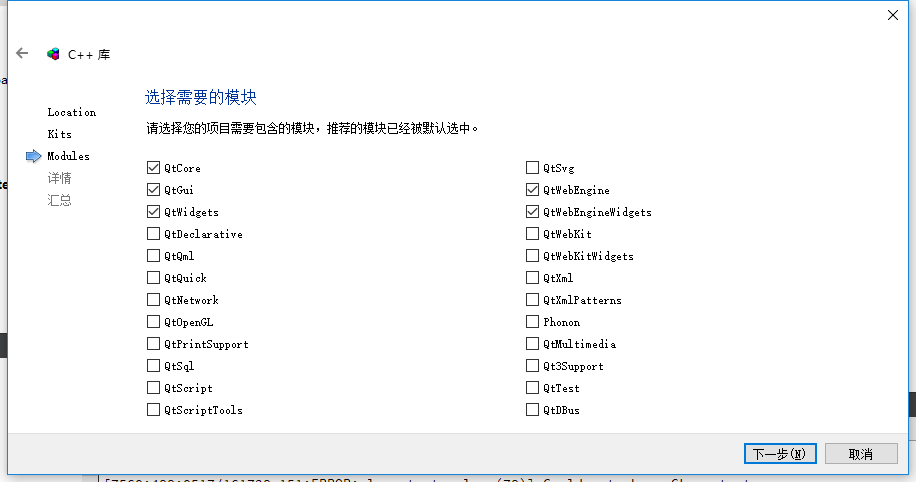
void enableFindButton(const QString &text);

* 1. 信号emit findPrevious(text, cs);
  2. 槽函数：需要在GUI里添加；
  3. connect(ui->pushButton\_2,SIGNAL(clicked()),this, SLOT(onResizeEcharts()));
  4. 一个信号可以连接多个槽函数，这些槽会一个接一个的被调用，但是它们的调用顺序是不确定的。

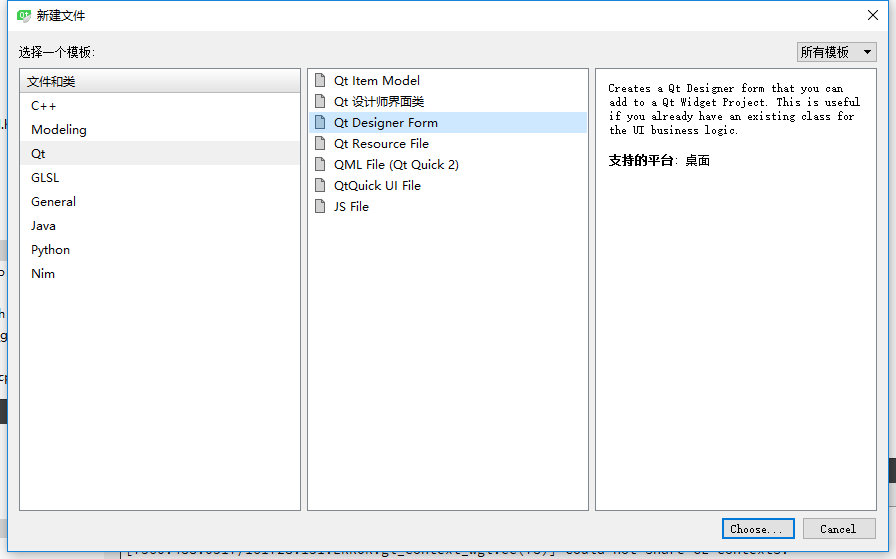
1. 事件
2. Shared library
   1. 创建动态链接库
      1. 创建Library项目

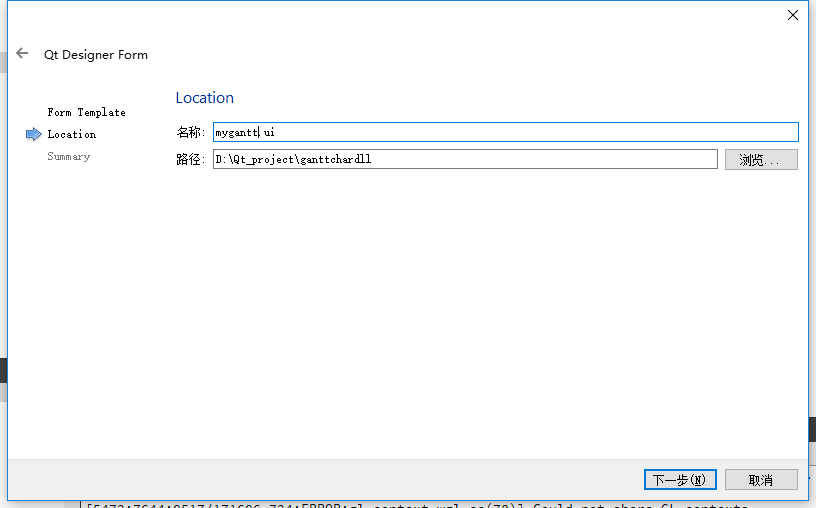


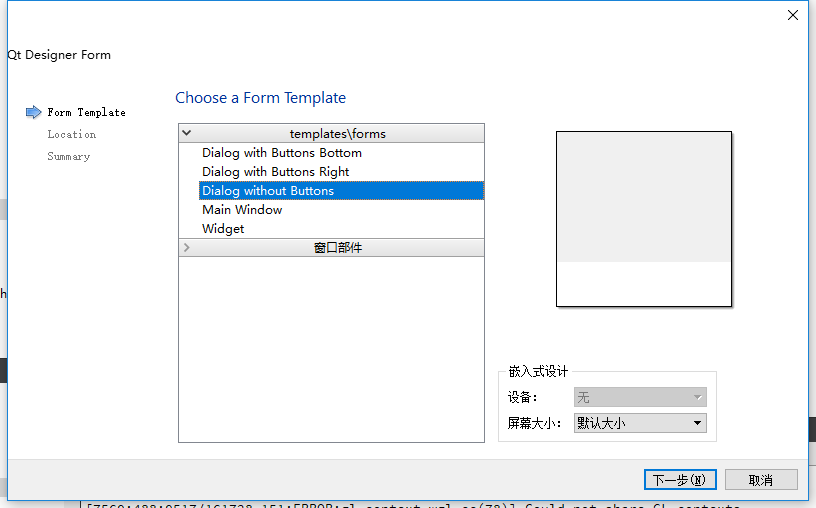




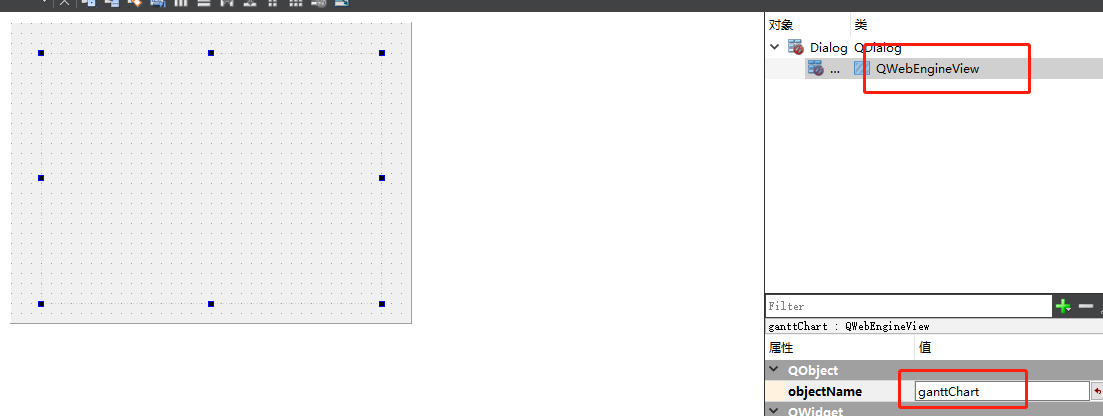
* + 1. 创建UI文件



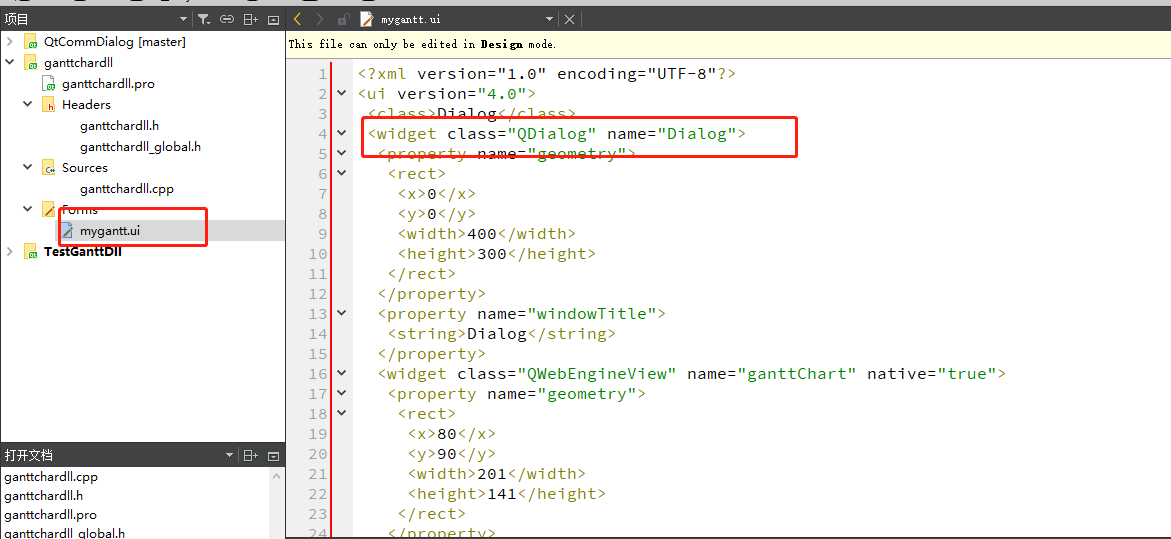


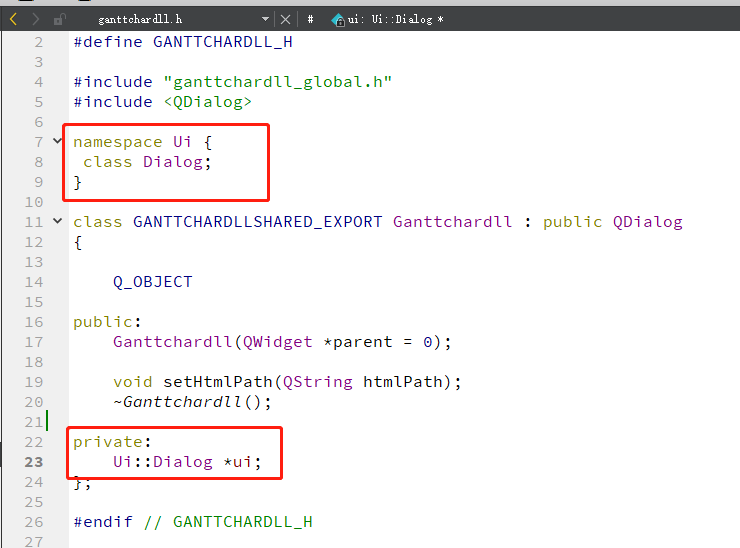


* + 1. 编辑UI文件，添加widget 提升为QWebEngineView;



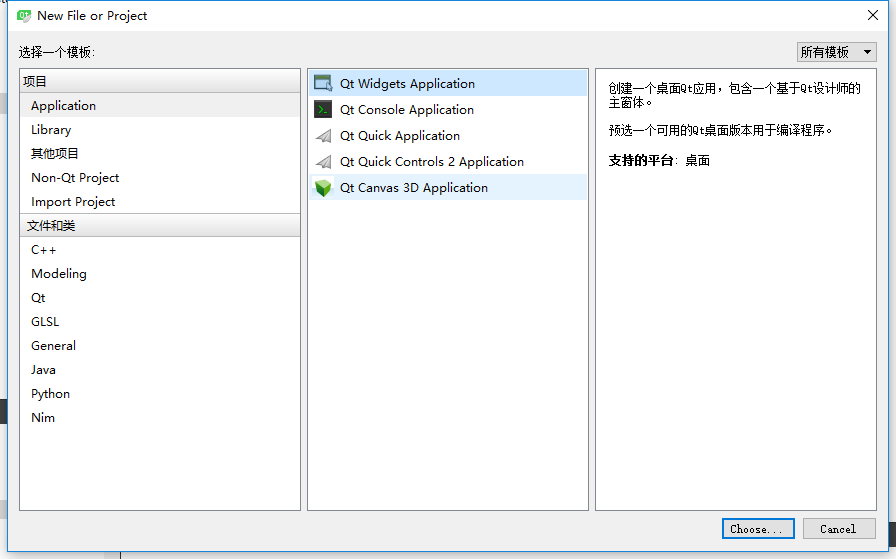
* + 1. 根据UI文件编辑头文件和源文件（将代码和UI关联，然后在代码控制UI控件）

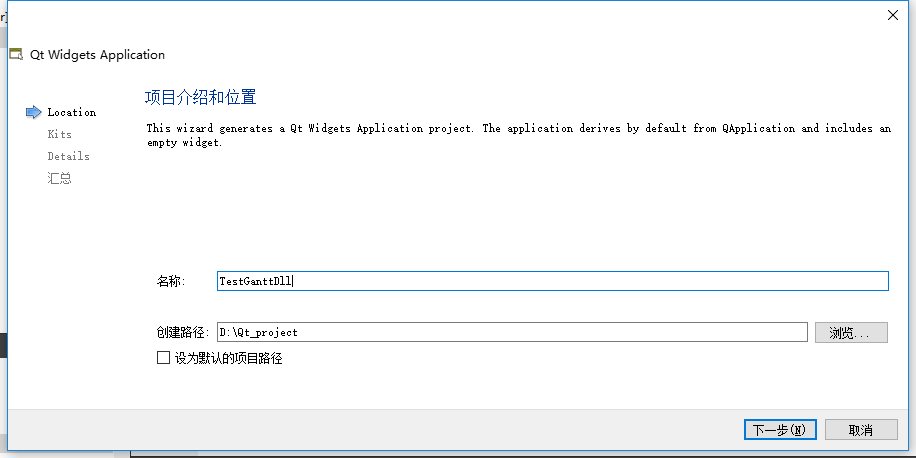




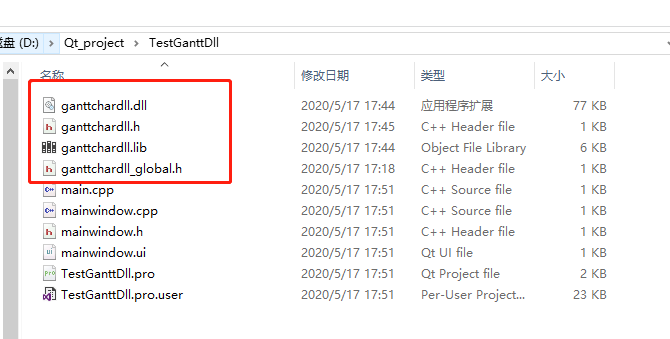


* + 1. 待定
  1. 使用动态链接库
     1. 创建项目



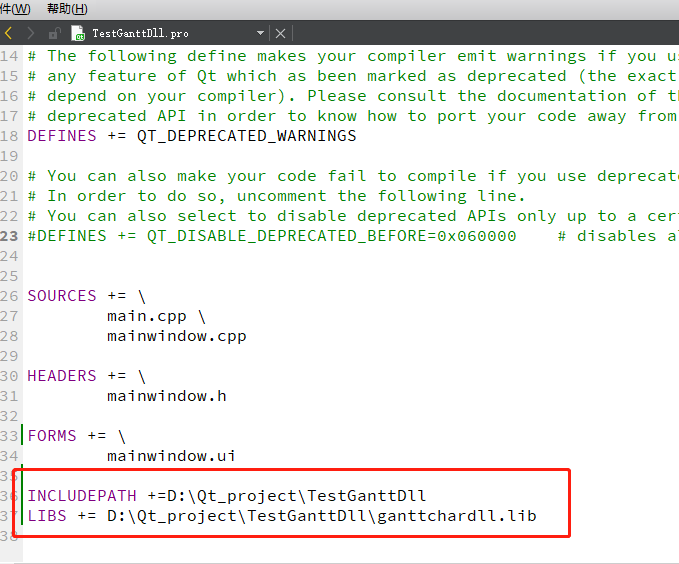


* + 1. 将动态链接库的四个文件拷贝到项目目录下

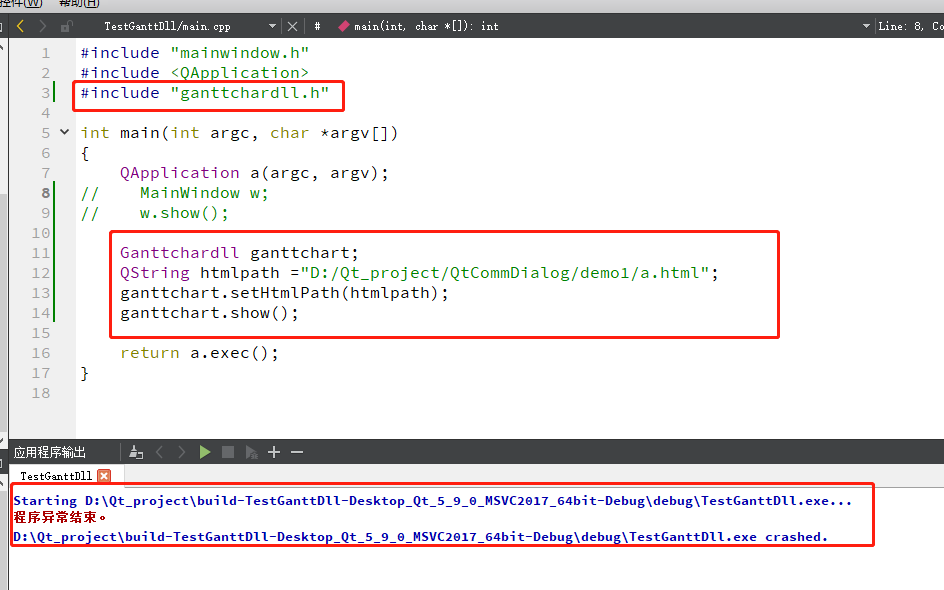


* + 1. 编辑项目的pro文件

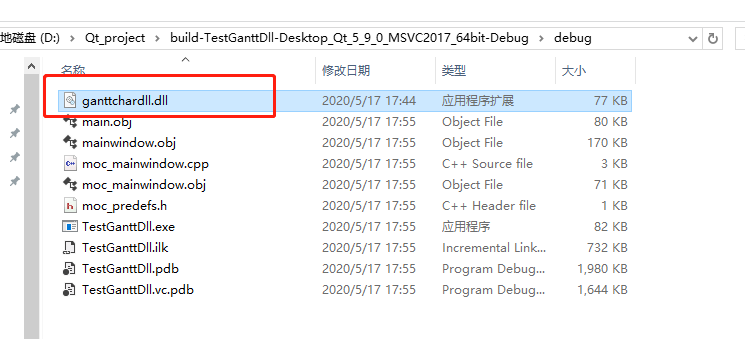
INCLUDEPATH 这一行就是为了让我们的项目可以找到 ganttchardll.h 和 ganttchardll\_global.h 这两个文件，你需要把这里的路径替换成符合你的工程的路径。LIBS 这一行则需要告诉编译器（注意，这里是编译器）到哪里去找到这个 dll 文件。



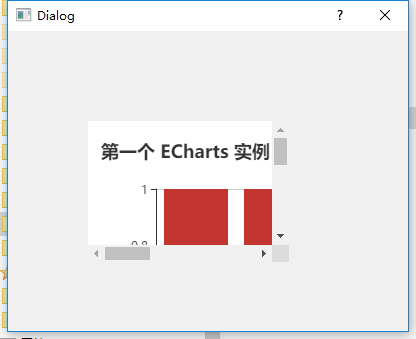
* + 1. 编辑源文件



* + 1. 找到工程下的exe文件，运行缺少dll,将dll复制到当前文件；（运行时需要dll,查找顺序：当前路径-系统路径）



* + 1. 运行即可



* + 1. 待定
  1. 更新dll

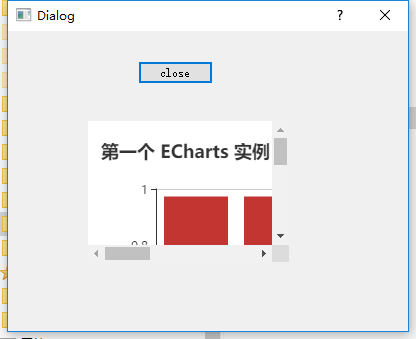
共享库的一个好处是可以动态加载，也就是说，如果你需要升级程序，那么就要简单的替换掉这个 dll 就好了，不需要要求用户重新安装全部文件。当然，这些 dll 也是有缺点的：动态加载的东西肯定会比静态编译的东西效率低一些。不过在现在的硬件环境下，这点性能损失已经可以忽略不计了。

1、修改共享库的代码，添加一个按钮，槽函数为点击关闭。

2、选择重新构建项目

3、将项目目录下的dll替换新的dll。

4、运行即可



* 1. 待定

1. 参考文献

<https://www.w3cschool.cn/learnroadqt/tufx1j3q.html>

<http://c.biancheng.net/view/1824.html>

<https://blog.csdn.net/octdream/article/details/90482654>

<https://blog.csdn.net/tennysonsky/article/details/48030333>

<https://blog.csdn.net/q496713258/article/details/6990837>

1. 待定