【文本编辑器】五、“帮助”功能实现

原创 Qt 学习 Qt 学习 2020-02-27

首先，基于 Qt 提供的标准消息对话框，通过几行代码实现了“关于”和“关于 Qt”消息框的简单功能；然后，讲解了根据用户需求设计的部分自定义消息框和完全自定义对话框的实现；最后一部分为设置应用程序图标和主界面字体。

应用程序【文本编辑器】的完整源代码可在后台回复「**文本编辑器**」获得下载链接。

本篇目录

运行环境：

win 10 + Qt 5.12.5 + Qt Creator 4.10

1. “帮助”主菜单栏实现

在 "textedit.h" 中添加“关于”功能的槽函数声明

```
1  private slots:
2      void about()
```

实现代码如下

```
1  void TextEdit::about()
2  {
3      QString str = tr("该示例演示了 Qt 文本编辑功能的应用，\n"
4                       "并提供了示例文档供您试用。\n\n"
5                       "想了解并获取的更多，欢迎关注微信公\n"
6                       "众号：Qt 学习 (ID: Qt_Learning )。");
7      QMessageBox::about(this, tr("关于"), str);
8  }
```

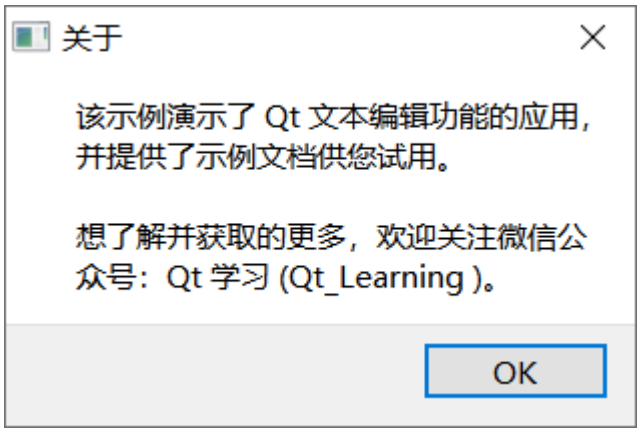
其中，`QMessageBox::about(this, tr("关于"), str)` 函数第一个参数为父窗口指针，第二个参数为消息框标题，第三个参数为消息框显示的文本提示。

添加“关于”和“关于 Qt”动作的 `connect` 函数

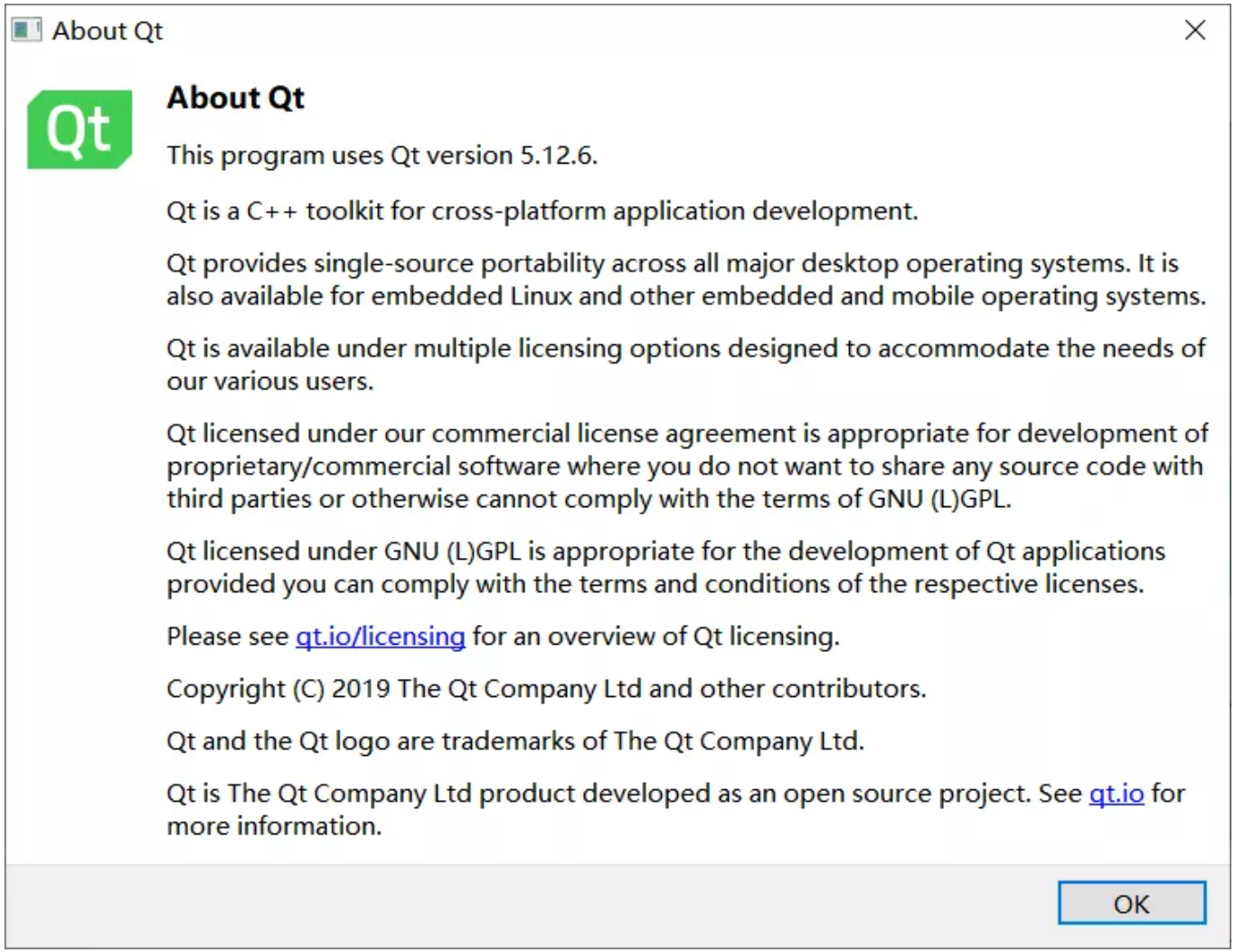
```
1   void TextEdit::setupHelpActions()
2   {
3       actionAbout = new QAction(tr("关于(&A)..."), this);
4       actionAbout->setStatusTip(tr("关于\"文本编辑器\""));
5       connect(actionAbout, &QAction::triggered, this, &TextEdit::about);
6
7       actionQtAbout = new QAction(tr("关于 Qt(&Q)..."), this);
8       actionQtAbout->setStatusTip(tr("关于 \"Qt\""));
9       connect(actionQtAbout, &QAction::triggered, qApp, &QApplication::aboutQt);
10
```

```
11          ...
12      }
```

运行程序，在『帮助』主菜单下分别点击『关于』和『关于 Qt』子项，弹出的消息对话框分别如下：



关于文本编辑器



关于 Qt

2. 自定义帮助对话框

上一节中的标准 About 消息框如若不能满足开发需求，也可选择自定义“关于”消息框，包括部分自定义消息框和完全自定义对话框两种，下面其设计和使用方法。

第一种为部分自定义“关于”消息框，重写 "textedit.cpp" 文件中的 `about()` 函数，其代码如下

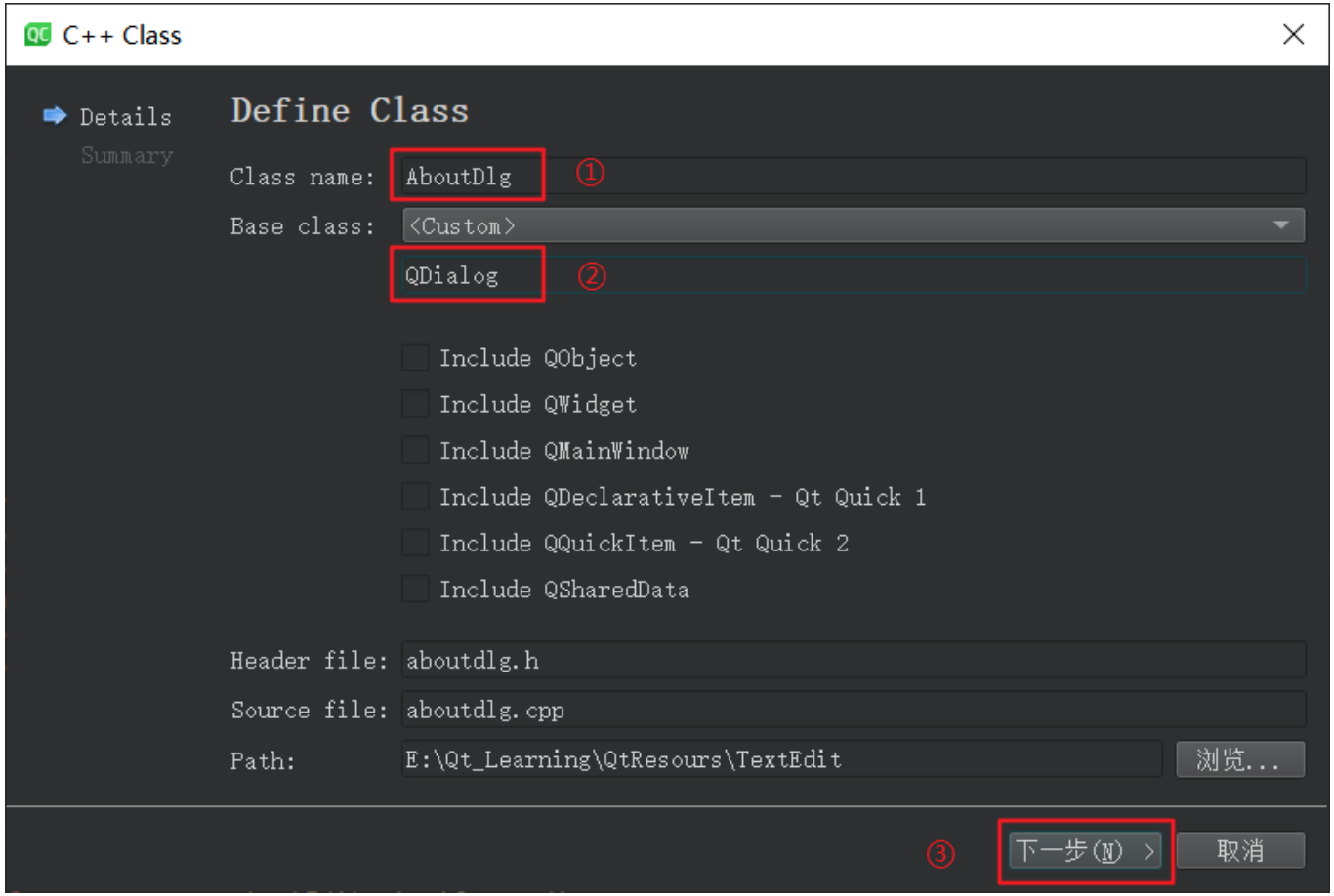
```
void TextEdit::about()
{
    QString str = tr("该示例演示了 Qt 文本编辑功能的应用，\n"
                     "并提供了示例文档供您试用。\n\n"
                     "想了解并获取的更多，欢迎关注微信公\n"
                     "众号：Qt 学习 (ID: Qt_Learning )。");

    QMessageBox customMsgBox;
    customMsgBox.setWindowTitle(tr("自定义帮助对话框"));
    customMsgBox.addButton(QMessageBox::Ok); // 设置自定义帮助对话框的按钮
    customMsgBox.setText(str);               // 设置自定义帮助对话框中显示的内容
    QPixmap pix(rsrcPath + "/Qt_logo.png");  // 下载源代码，其中包含此图片资源
    customMsgBox.setIconPixmap(pix.scaled(QSize(128, 128))); // 设置自定义帮助对话框图标
    customMsgBox.exec();                                     // 显示此自定义帮助对话框
}
```

运行程序，弹出的“关于”消息框效果如下

第二种为完全自定义“关于”对话框，其设计实现过程如下：

在项目 "TextEdit" 上右击，单击『Add New...』，在模板中选择『C++』和『C++ Class』，选择之后弹出另一对话框，输入类名称 "AboutDlg" 和基类 "QDialog"，如下图，单击『下一步』完成向项目中添加新类



完善 "aboutdlg.h" 文件中的代码如下

```
1  #ifndef ABOUTDLG_H
2  #define ABOUTDLG_H
```

```
3
4   #include <QDialog>
5
6   class AboutDlg : public QDialog
7   {
8       Q_OBJECT
9
10  public:
11      AboutDlg(QWidget* parent = nullptr);
12  };
13
14  #endif // ABOUTDLG_H
```

完善 "aboutdlg.cpp" 文件中的代码如下

```
1   #include "aboutdlg.h"
2   #include <QLabel>
3   #include <QPushButton>
4   #include <QVBoxLayout>
5   #include <QHBoxLayout>
6
7   AboutDlg::AboutDlg(QWidget* parent)
8       : QDialog(parent)
9   {
10      setWindowTitle(tr("自定义帮助对话框"));
11      setWindowIcon(QIcon(":/myImages/images/Qt_logo.png"));
12
13      QVBoxLayout *layout1 = new QVBoxLayout;
14      QLabel *iconLabel = new QLabel;
15      QPixmap pix1(":/myImages/images/Qt_logo.png");
16      iconLabel->setPixmap(pix1.scaled(QSize(128, 128)));
17      QLabel *idLabel = new QLabel(tr("ID: Qt_Learning"));
18      QLabel *qrLabel = new QLabel;
19      QPixmap pix2(":/myImages/images/Qt_QR.jpg");
20      qrLabel->setPixmap(pix2.scaled(QSize(128, 128)));
21      QLabel *label1 = new QLabel(tr("扫描二维码关注"));
22      layout1->addWidget(iconLabel, 0, Qt::AlignHCenter);
```

```
23      layout1->addWidget(idLabel, 0, Qt::AlignHCenter);
24      layout1->addSpacing(20);
25      layout1->addWidget(qrLabel, 0, Qt::AlignHCenter);
26      layout1->addWidget(label1, 0, Qt::AlignHCenter);
27
28      QLabel *labels[7];
29      QString names[7];
30      names[0] = "<a style='color:green;' href=\"https://mp.weixin.qq.com/s/HyemCP79yZqjLF5WSagCrQ\">界面设计";
31      names[1] = "<a style='color:green;' href=\"https://mp.weixin.qq.com/s/frPJymEyV5tRpqlgjfcbDQ\">文件操作功能（上）";
32      names[2] = "<a style='color:green;' href=\"https://mp.weixin.qq.com/s/tIjz2l-xZOAgfTJwQUxQTg\">文件操作功能（下）";
33      names[3] = "<a style='color:green;' href=\"https://mp.weixin.qq.com/s/V4Hm31LLwusG278XolBm6w\">文本编辑功能";
34      names[4] = "<a style='color:green;' href=\"https://mp.weixin.qq.com/s/h_sMAqNpF9nPbZob5crTgg\">文档排版美化功能（上）";
35      names[5] = "<a style='color:green;' href=\"https://mp.weixin.qq.com/s/-ye99b9vdaiomCjYZBeqXg\">文档排版美化功能（下）";
36      names[6] = "<a style='color:red;' href=\"https://mp.weixin.qq.com/s/zXnKW3d7t04ABIrV1gr0cg\">功能完善";
37
38      QVBoxLayout *layout2 = new QVBoxLayout;
39      QString str = tr("该示例演示了 Qt 文本编辑功能的应用。\n"
40                       "并提供了示例文档供您试用。\n\n"
41                       "想了解并获取的更多，欢迎关注微信公\n"
42                       "众号：Qt 学习 (ID: Qt_Learning)。");
43      QLabel *aboutLabel = new QLabel(str);
44      QLabel *label2 = new QLabel(tr("【文本编辑器】系列："));
45      layout2->addWidget(aboutLabel);
46      layout2->addStretch();
47      layout2->addWidget(label2);
48      for (int i = 0; i < 7; i++) {
49          labels[i] = new QLabel;
50          labels[i]->setOpenExternalLinks(true);
51          labels[i]->setText(names[i]);
52          layout2->addWidget(labels[i]);
53      }
54      layout2->addStretch();
55
56      QHBoxLayout *layout3 = new QHBoxLayout;
57      layout3->addLayout(layout1);
58      layout3->addSpacing(30);
59      layout3->addLayout(layout2);
60
61      QVBoxLayout *mainLayout = new QVBoxLayout(this);
```

```
62      QPushButton *button = new QPushButton(tr("Ok"));
63      QHBoxLayout *layout4 = new QHBoxLayout;
64      layout4->addStretch();
65      layout4->addWidget(button);
66      connect(button, SIGNAL(clicked()), this, SLOT(accept()));
67      mainLayout->setMargin(20);
68      mainLayout->addLayout(layout3);
69      mainLayout->addLayout(layout4);
70  }
```

此对话框的设计不难，具体可参考《代码化 UI 设计》一文。额外说明下，30~36行代码设置标签的超链接，其中

- `<>` 内的字符包含两部分：
  `style='color:green;'` 设置超链接文本颜色
  `about(href=\"...\")` 设置超链接地址
- `<>` 外的字符为标签的显示字符

为在程序主界面点击『帮助』『关于』时能够弹出自定义对话框，在 "textedit.cpp" 文件中添加声明

```
1   #include "aboutdlg.h"
```

并重写 "textedit.cpp" 文件中的 `about()` 函数，其代码如下

```
1   void TextEdit::about()
2   {
3       AboutDlg *aboutDlg = new AboutDlg();
4       aboutDlg->show();
5   }
```

运行程序，完全自定义对话框效果如下

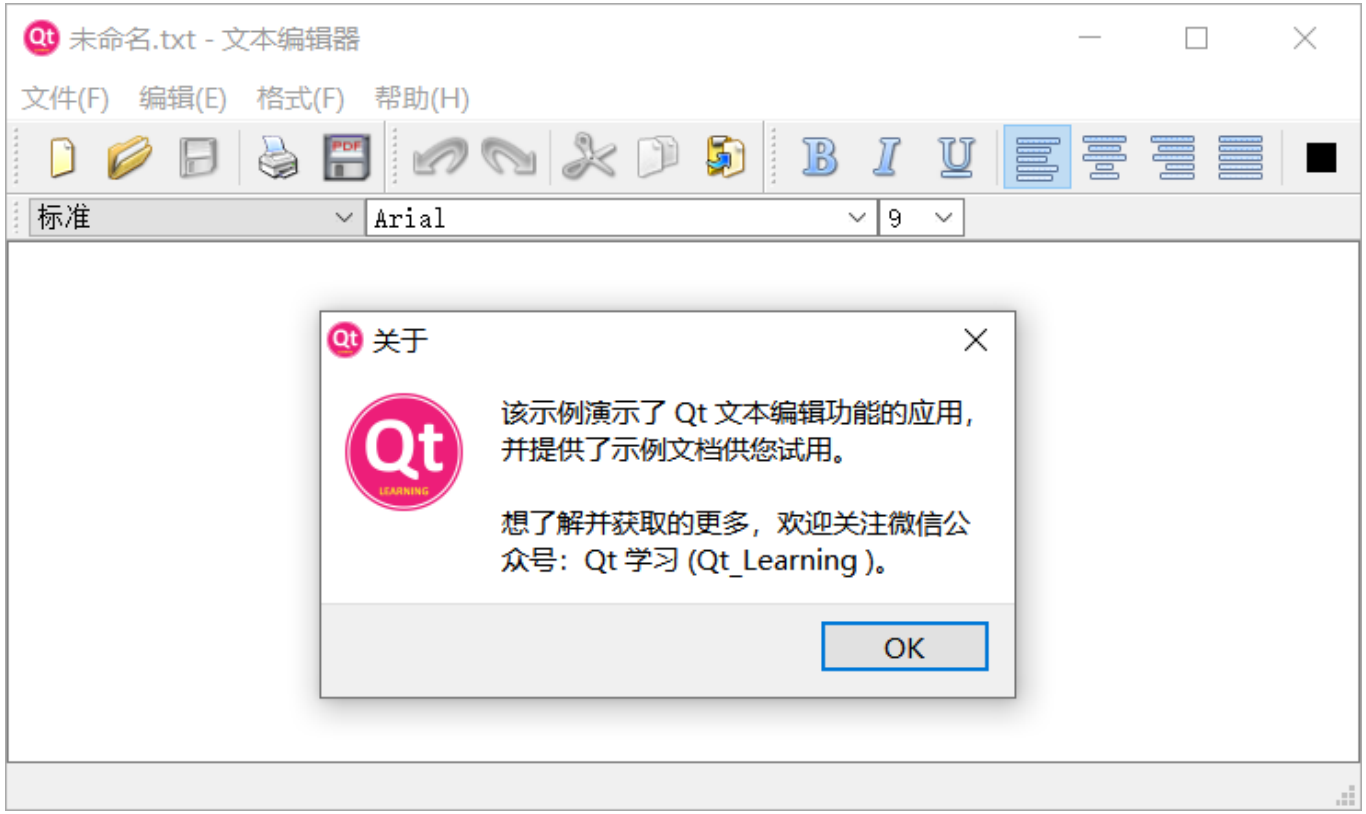
单击绿色字体可打开相应链接

3. 完善程序组界面

在 "textedit.cpp" 文件的构造函数中添加设置程序图标代码如下

```
1   setWindowIcon(QIcon(rsrcPath + "/Qt_logo.png"));
```

以本篇第 1 节的 `about()` 函数为例，设置应用程序图标后的主界面效果和“关于”消息框效果如下



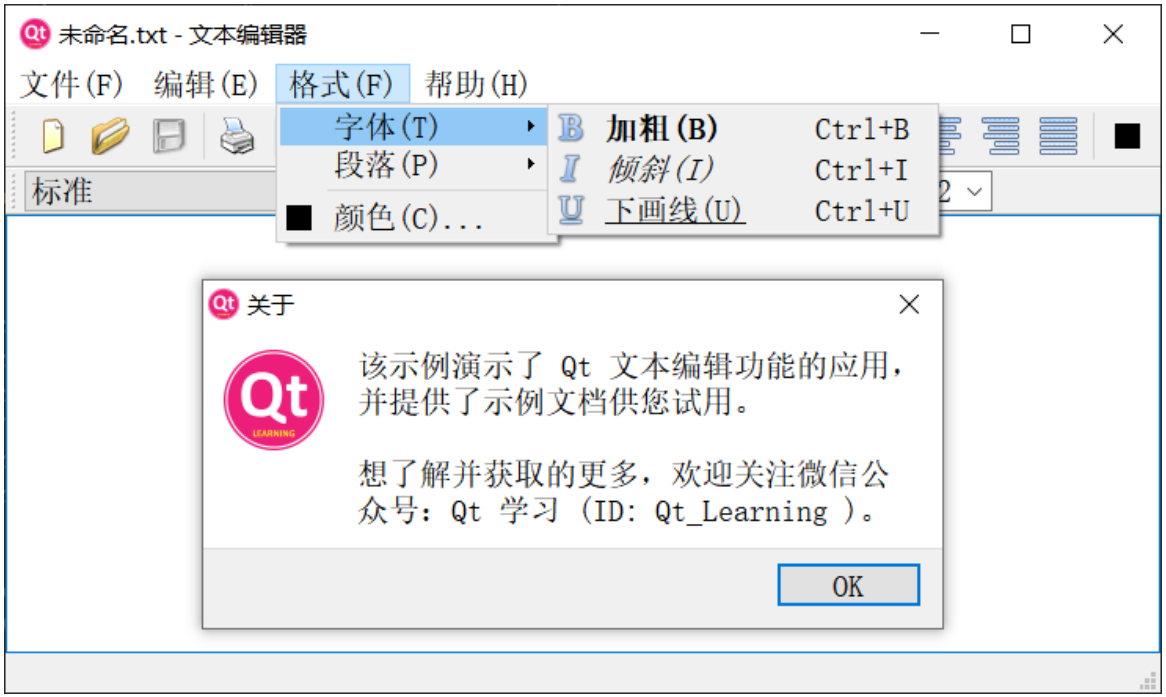
可以看到程序主界面左上角的图标，以及“关于”消息框将应用程序的图标也添加在内。

打开 "main.cpp" 文件，添加界面字体设置代码（加粗部分代码）如下

```
1  #include "textedit.h"
2
3  #include <QApplication>
4
5  int main(int argc, char *argv[])
```

```
6   {
7       QApplication a(argc, argv);
8       QFont font("ZYSong18030", 12);
9       a.setFont(font);
10      TextEdit w;
11      w.show();
12      return a.exec();
13  }
```

运行程序，主界面字体、菜单栏子项字体以及弹出的对话框字体效果如下



小结

使用 Qt 提供的标准消息对话框，通过几行代码实现了“关于”和“关于 Qt”对话框功能；根据用户需求实现了部分自定义对话框和完全自定义对话框，如自定义显示的图片和链接功能的标签；为应用程序设置图标和主界面字体。

【文本编辑器】序列以此篇结束，谢谢大家。

应用程序【文本编辑器】的完整源代码可在后台回复「**文本编辑器**」获得下载链接。

相关阅读：

《【文本编辑器】一、界面设计》
《【文本编辑器】二、文件操作功能（上）》
《【文本编辑器】二、文件操作功能（下）》
《【文本编辑器】三、文本编辑功能》
《【文本编辑器】四、文档排版美化功能（上）》
《【文本编辑器】四、文档排版美化功能（下）》

文章已于2020/02/27修改

喜欢此内容的人还喜欢

习近平给“国际青年领袖对话”项目外籍青年代表回信

中国政府网

金牌送女儿、奖金给老婆买包，没想到他是这样的世界冠军

她刊