**“C++程序设计与训练”课程大作业**

**项目报告**

**项目名称：**

**基于C++的购物系统程序**

**姓名：** 倪文宇

**学号：**2014010623

**班级：** 自45

**日期：**

目 录

[1系统功能设计 4](#_Toc430472618)

[1.1总体功能描述 4](#_Toc430472619)

[1.1.1总管理模块 4](#_Toc430472620)

[1.1.2销售统计模块 6](#_Toc430472621)

[1.1.3登陆注册模块 6](#_Toc430472622)

[1.1.4错误报告模块 6](#_Toc430472623)

[1.2功能点说明 6](#_Toc430472624)

[2系统总体结构 8](#_Toc430472625)

[2.1概要设计 8](#_Toc430472626)

[2.1.1可视化界面-数据库-图表 8](#_Toc430472627)

[2.1.2数据库与程序的交互 8](#_Toc430472628)

[2.2小组分工 9](#_Toc430472629)

[3本人工作内容 9](#_Toc430472630)

[3.1界面设计模块 9](#_Toc430472631)

[3.1.1登陆注册界面 9](#_Toc430472632)

[3.1.2错误反馈界面 10](#_Toc430472633)

[3.2主、子界面连接 12](#_Toc430472634)

[3.2.1按钮（Pushbutton）连接 12](#_Toc430472635)

[3.2.2菜单条（Menu Bar）连接 13](#_Toc430472636)

[3.2.3基于QtDesigner的设计 13](#_Toc430472637)

[3.3数据输入模块 14](#_Toc430472638)

[3.3.1组合框Combo Box 14](#_Toc430472639)

[3.3.2输入文本框Line Edit 15](#_Toc430472640)

[3.3.3按钮Pushbutton 15](#_Toc430472641)

[3.4数据输出模块 15](#_Toc430472642)

[3.4.1表视图的应用Table View 15](#_Toc430472643)

[3.5数据库管理 16](#_Toc430472644)

[3.5.1数据库建立 16](#_Toc430472645)

[3.5.2操作数据 17](#_Toc430472646)

[3.6界面美化 18](#_Toc430472647)

[3.6.1程序启动界面 18](#_Toc430472648)

[3.6.2插入图片 18](#_Toc430472649)

[3.6.3修改图标 19](#_Toc430472650)

[3.6.4界面大小的固定 19](#_Toc430472651)

[3.7关于容错 19](#_Toc430472652)

[3.7.1消息框的应用 19](#_Toc430472653)

[4项目总结 20](#_Toc430472654)

[4.1Qt静态编译的实现 20](#_Toc430472655)

[4.2中文乱码的解决 21](#_Toc430472656)

[4.3数据库中文乱码的问题 22](#_Toc430472657)

[4.4关于压缩 23](#_Toc430472658)

[4.5关于相关资料的搜寻和自学 24](#_Toc430472659)

[5相关问题的说明 24](#_Toc430472660)

# 1系统功能设计

## 1.1总体功能描述

### 1.1.1总管理模块

简洁美观的界面上有七个按钮，帮助使用者直观地打开对应的子窗口。

此外，在主窗体的子菜单栏提供切换界面的功能，例如修改密码界面和数据统计的界面。不同于按钮切换，子菜单栏可支持快捷键。

#### 1.1.1用户注册

窗体打开时，子窗体获取目前用户表中的所有用户信息。

实现用户管理，在文本框中输入一个用户的资料，按“添加”按钮，可以将新用户的信息加入用户表中；同时下边的子窗体被刷新。

单击“清空”按钮，把窗体上的文本框的内容全部清空，为下次输入做准备。

#### 1.1.2快递公司管理

窗体打开时，子窗体中获取目前快递公司表中的所有快递公司信息。

实现快递公司管理功能。在窗体上的文本框中输入一个快递公司的资料，按添加按钮，可以将快递公司的信息加入快递公司表中，同时更新子窗体中的数据；

点击窗体下方“待删除公司”下拉列表选择快递公司，按删除按钮，可以将该公司的信息从快递公司表中删除，同时更新子窗体中的数据；

#### 1.1.3商品管理

该窗体实现商品管理功能。窗体打开时，子窗体中获取目前商品表中的所有商品。在窗体上方的4个文本框中输入一个商品的资料，单击“添加商品”按钮，可以将商品信息加入商品表中，同时刷新子窗体； 单击“清空”按钮清空文本框内容，为下次输入做准备。

窗体下方的“选择商品”组合框中显示商品表中的商品编号、商品名称和库存量（商品按照商品编号从小到大排列），选择一种商品，在“设置库存数量”文本框中输入一个数目，点击“更改”按钮，更新商品表中该商品的库存数量，同时刷新子窗体。

#### 1.1.4购买商品

该窗体实现商品购买功能。在文本框中输入用户名，在组合框的下拉选单中选择一个要买的商品，在选择快递公司组合框中选择一个快递公司，单击“购买”按钮，完成购买操作。

#### 1.1.5条件查询

在模糊查询前的输入框中输入想要查找的商品的关键字，如“键盘”，对商品进行模糊查找。

查询用户所有订单组合框中显示所有用户信息，选择一个用户，单击“精确查询”命令按钮，查找该用户的所有订单。

#### 1.1.6修改订单状态

选择快递公司组合框中显示所有的快递公司信息，选择订单编号组合框中显示上面选择的快递公司还没有配送的订单。

### 1.1.2销售统计模块

该窗体中实现显示快递销售的柱状图排名。

柱状图直观地给使用者选择的一个参考，让使用更合理和人性化。

### 1.1.3登陆注册模块

打开主窗体前有一个登录界面，输入正确密码方可登录。

密码修改界面支持修改默认密码，需输入旧密码以及两遍相同的新密码方可修改成功。

### 1.1.4错误报告模块

打开错误报告界面，可通过按钮前往GitHub该项目网页，进行错误反馈，以便更好优化。

## 1.2功能点说明

大作业功能点说明（示例）

|  |  |  |  |
| --- | --- | --- | --- |
| **功能类型** | **功能点名称** | **实现方式**   1. **自己编写C++代码** 2. **使用C++标准库** 3. **使用第三方库** 4. **使用SQL语句** | **功能点描述** |
| 基本信息处理 | 商品信息的读写 | 2 | 通过标准库函数支持读入和输出文本，处理大量数据 |
| 商品、订单、物流信息增加 | 4 | 用户可在输入框中输入，并点击按钮增加信息 |
| 商品、订单、物流信息删除 | 4 | 用户可在输入框中输入，并点击按钮删除信息 |
| 订单状态变更 | 4 | 查询并在下拉单中选中订单，点击按钮更改订单状态 |
| 订单简单查询 | 1 | 根据用户名直接查询该用户所有订单信息 |
| 容错特性 | 1 | 提示用户输入错误，使程序正常运行 |
| 数据库支持 | 数据库读写 | 4 | 连接程序和数据库，实现可利用SQL语句控制 |
| 网络连接支持 | 支持连接到网络 | 3 | 支持连接到GitHub网页，反馈软件问题 |
| 复杂查询功能 | 排序 | 1 | 实现根据价格和销售热度的升降序排序 |
| 模糊查询 | 4 | 根据用户输入关键字进行相关查询 |
| 统计功能 | 统计销量 | 3 | 对所有销售数据进行统计，统计出相应的销售量 |
| 其它功能 | 绘制图表 | 3 | 对快递销售统计绘图，绘制出销售量柱状图。 |

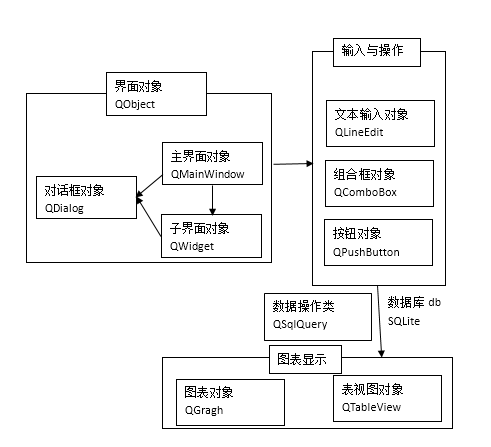
# 2系统总体结构

## 2.1概要设计

对于购物系统，我们想通过数据库来实现，这样可以将多类数据集成在一个数据库中，且利用数据本身的加密功能，实现良好的数据封装。所以对于这样一个系统，主要分为三个主要的架构，分别为可视化界面-数据库-图表，通过可视化界面给用户更为直观便捷地操作数据库，同时通过图表直观反映数据，从而实现应用程序与数据间的交互。

### 2.1.1可视化界面-数据库-图表

主要包含三个类分别为界面类，输入与操作类、图表类，其中的关系如图：



### 2.1.2数据库与程序的交互

在输入与操作类中通过类中的函数读取数据，并通过QSqlQuery中的exec成员函数利用Sql语句对数据实现“添加”，“修改”，“删除”，“精确查询”，“模糊查询”

## 2.2小组分工

表1 小组成员分工说明

|  |  |  |  |
| --- | --- | --- | --- |
| 小组成员姓名 | 小组成员班级 | 小组成员学号 | 小组成员分工 |
| 倪文宇 | 自45 | 2014010623 | * 各类对象的绘制与管理，类的设计 * 主框架、子框架、应用程序间的数据交互，数据库的建立和数据操作 * 用户登录注册、错误反馈界面的编写 * 界面美化 |
| 柳荫 | 自45 | 2014011858 | * 主体框架的搭建 * 系统和数据库的连接 * 数据统计柱状图的实现 |

# 3本人工作内容

## 3.1界面设计模块

### 3.1.1登陆注册界面

作为一个购物系统，为操作该购物系统的使用者（管理员）需要有权限区分，登录需要检验密码，密码正确方可登录。考虑到这个问题，添加了LoginDialog类，在打开主界面前，程序启动界面后添加了一个管理员登录界面，如下图：



对于密码的管理是在数据库中建立了一个Password表，其中存储管理员的密码，且要在主界面显示之前实现，在主调函数中添加修改了如下代码：

LoginDialog dlg;

if(dlg.exec()==QDialog::Accepted){

w.show();//当密码正确时，调用主界面w显示

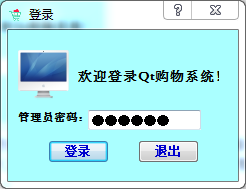
return a.exec();

}else{

return 0;//当密码不正确且退出，程序结束

}

另一方面，考虑到用户的隐私和安全性，对与输入框实行了加密，修改echoMode为Password，效果如下图：



使得用户输入的密码不会被周围的人看到而泄密。

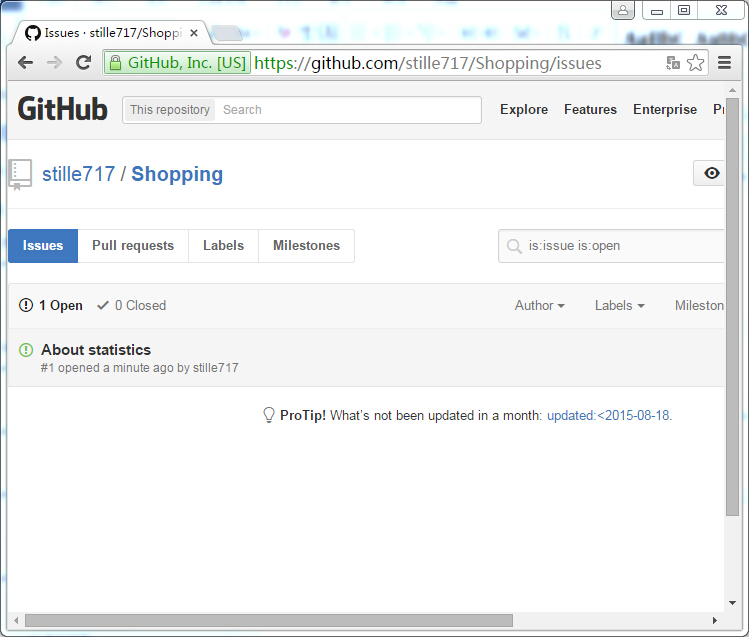
### 3.1.2错误反馈界面

为了更好的用户体验，我们希望用户在发现任何使用上的不便和软件上的bug都能即使反映给我们，因此在GitHub上建立了该项目，并建立连接，用户可方便地通过按钮一键连接，将问题反馈给我们。

界面设计如下：



点击“Contact Us”，转到GitHub该项目的网页：



添加反馈issues，具体描述遇到的问题或不便。

对于网页连接原先的考虑是利用Qt的webview类将此界面作为一个简易浏览器，浏览网页，但实际操作时发现，windows对于第三方的浏览器有限制，且需要设置network，用户使用较为麻烦，且浏览效果也不好，因此考虑到市面上许多应用软件对于该类用户反馈的实现均为弹出系统本地浏览器，浏览固定网页。

对此的实现利用QDesktopServices中的OpenUrl实现：

QDesktopServices::openUrl(QUrl(QLatin1String("https://github.com/stille717/Shopping.git")));

## 3.2主、子界面连接

由于界面较多，且使用时并非所有都立即需要用到，因此对于不同的界面需要连接，连接的方式主要通过按钮连接和菜单条连接两种

### 3.2.1按钮（Pushbutton）连接

按钮的连接主要用于主菜单和子菜单之间的连接，且这样连接的子界面使用时不会有互相之间的影响

以“用户注册”界面为例，在MainWindow.cpp中包含头文件：

#include "login.h"

#include "ui\_login.h"

在对应的按钮的槽函数中，定义login界面，并显示

void MainWindow::on\_loginButton\_clicked()

{

Login \*ui=new Login;//建立

ui->show();//显示

}

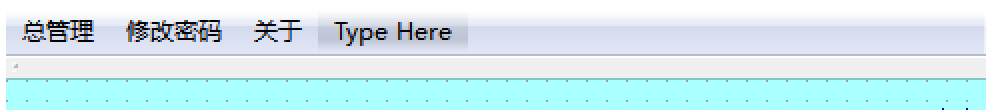
效果即如下图：



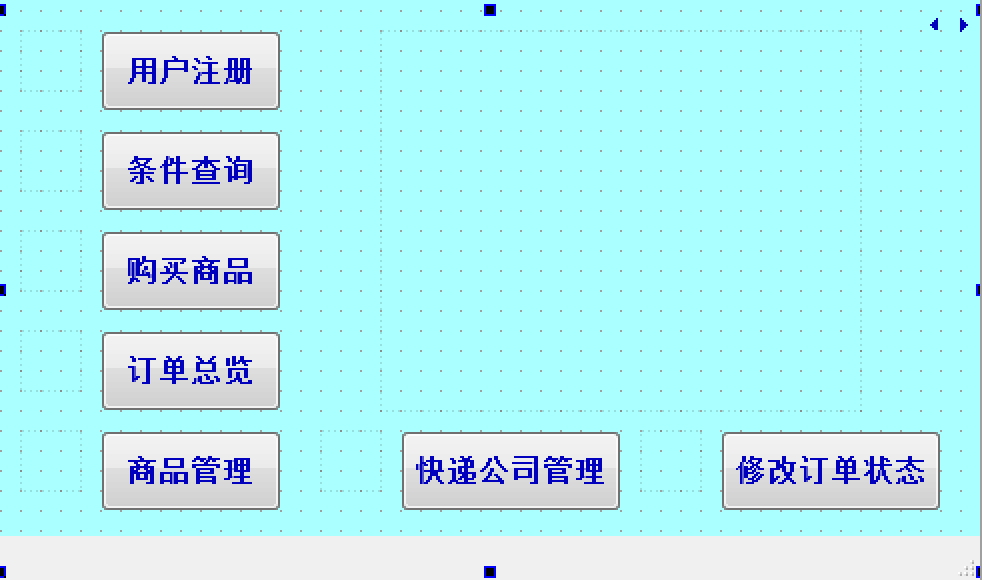
### 3.2.2菜单条（Menu Bar）连接

对于另一些相较而言并不常用而又必须的功能，考虑使用Menu Bar切换，例如“密码修改”是必须的，但在使用时并非主要功能，类似的还有“错误反馈”、“数据统计”。

设计时在Main Window上添加



将不同窗体的内容放在stacked Widget的不同页中



再在action的trigger函数中设定当前主页current Index，例如：

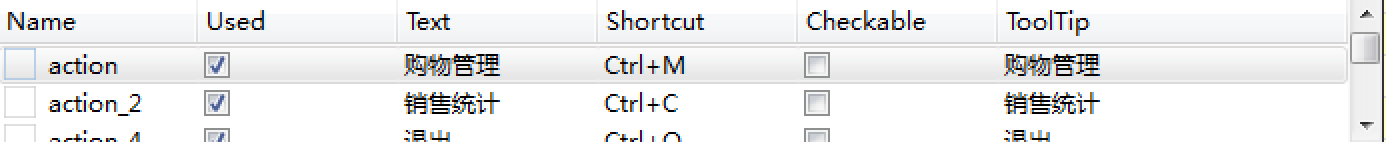
void MainWindow::on\_action\_triggered()

{

ui->stackedWidget->setCurrentIndex(0);

}

此外，对于不同的action可以添加相应的快捷键，如下图所示：



### 3.2.3基于QtDesigner的设计

Qt虽然提供了界面全代码编写，但此次编写并没有完全采用，而是选择了用QtDesigner编写界面ui文件，主要是考虑到编写时项目的设计还没有完全完成，在后续的修改中可能会有较大的变动，如按钮的移动、MenuBar和StackedWidget的组合，如果使用全代码编写（重新计算水平和竖直排版，转移的窗口部件全重新编写），这些修改将相当麻烦。

此外对于信号与槽函数的处理也会方便很多，只需在对应部件的槽函数进行修改，方便明确。

## 3.3数据输入模块

对于数据的输入和读写，采用了综合利用组合框、输入文本框和按钮的方法。

### 3.3.1组合框Combo Box

#### 3.3.1.1动态添加

由于需要支持用户自己添加的用户信息、商品信息以及快递信息，因此在下拉框中的items需要支持动态添加，而非静态。先从数据库中用QSqlQuery读取相关信息，并用QSqlQueryModel添加到ComboBox中，代码如下：

QSqlQueryModel \*umodel=new QSqlQueryModel(this);

umodel->setQuery(QString("select name from user"));

ui->ucomboBox->setModel(umodel);

#### 3.3.1.2自动补全

考虑到商品信息可能较多，在商品管理界面的下拉框中实现了自动补全功能，即输入部分商品名称，就会显示相应可能的商品信息，方便使用者选择，效果如下图：



自动补全功能的实现利用了QCompleter类，用QStringList添加Items：

QStringList strings;

QSqlQuery query("select sname from commodity");

while (query.next()) {

QString sname = query.value(0).toString();

strings.append(sname);

}

QCompleter \*completer = new QCompleter(strings, this);

ui->scomboBox->clear();

ui->scomboBox->addItems(strings);

ui->scomboBox->setCompleter(completer);//实现自动补全功能

### 3.3.2输入文本框Line Edit

利用ui->lineEdit->text()即可获得输入文本框中的内容，例如登录界面的输入：

if(pwd==ui->lineEdit->text()){

QDialog::*accept*();

}//判断输入与密码是否一致

同时，由于text()的输入数据类型为String，可调用toInt()等函数转成所需的输入数据。

### 3.3.3按钮Pushbutton

在按钮的槽函数slot中添加对于数据的读取和处理，例如“用户注册”界面对于用户数据的添加：

void Login::on\_pushButton\_clicked()

{…

读取用户的数据并处理

…}

## 3.4数据输出模块

对于数据的输出，主要是基于数据库中的表，而TableView对于表的输出是一个很好的选择。

### 3.4.1表视图的应用Table View

对于表中数据的大量读取和输出，需要用到QSqlTableModel，查询到所需的数据，设置好表头，setmodel到tableview即可，代码如下：

QSqlTableModel \*model= new QSqlTableModel;

model->*setTable*("user");//选择表

model->*setEditStrategy*(QSqlTableModel::OnManualSubmit);

model->*select*();

model->*setHeaderData*(0, Qt::Horizontal, QObject::tr("id"));//设置表头

model->*setHeaderData*(1, Qt::Horizontal, QObject::tr("name"));

model->*setHeaderData*(2, Qt::Horizontal, QObject::tr("address"));

model->*setHeaderData*(3, Qt::Horizontal, QObject::tr("telephone"));

ui->tableView->*setModel*(model);//输出到tableView

界面中的效果如下：



## 3.5数据库管理

数据库是本项目中相当重要的一环，而我负责的是数据库的建立和数据的操作。

### 3.5.1数据库建立

对于数据库的建立，这里参考的是《Qt5开发及实例》中的“汽车管理系统”中对数据库建立的操作。添加connection.h头文件，在其中createdconnection函数中用QSqlQuery对新建的数据库中添加表格和数据，如图：



编译一遍后即会生成data.db文件，其中包含了所需的数据：



这样生成的数据库是默认使用了Qt内置的QSQLite数据库，经测试和实验，相比于利用驱动QODBC读取微软的mdb数据库，有更好的稳定性。

### 3.5.2操作数据

在这里操作数据是使用QSqlQuery，并利用exec函数直接调用sql语句，简明直观。

#### 3.5.2.1添加

运用INSERT INTO table VALUES(…)语句，需要注意的是对于中文需要使用

query.exec(QObject::tr(“SQL语句”);（具体见4.3中解释）

例如用户注册中的添加：

QSqlQuery query;

bool correct = query.exec(QString("insert into user values('%1','%2','%3','%4') ").arg(ui->lineEdit->text()).arg(ui->lineEdit\_2->text()).arg(ui->lineEdit\_3->text()).arg(ui->lineEdit\_4->text()));//添加数据

if (correct) {

QMessageBox::information(this, "成功", "用户信息添加成功！", QMessageBox::Ok);

}

else {

QMessageBox::information(this, "失败", "添加失败，可能用户名已经存在！", QMessageBox::Ok);

}

#### 3.5.2.2修改

修改使用UPDATE语句，更新数据，例如在修改密码的代码：

bool correct=query.exec(QString("UPDATE password SET pwd='%1' WHERE pwd=='%2'").arg(ui->lineEdit\_3->text()).arg(pwOld));

#### 3.5.2.3删除

删除使用DELETE语句，删除数据，例如商品管理中删除商品：

bool correct=query.exec(QString("DELETE FROM commodity WHERE sname='%1' ").arg(ui->comboBox->currentText()));

#### 3.5.2.4模糊查询

模糊由于需要输出到TableView，需要先输出到QSqlTableModel，即：

QSqlTableModel\* model = new QSqlTableModel(this);

model->*setTable*("commodity");

QString text=ui->lineEdit->text();

model->*setFilter*(QObject::tr("sname like '%%%1%%'").arg(text));//实现模糊查询

其中"sname like '%%%1%%'"相当于SELECT \* FROM table LIKE ‘\*’&value& ‘\*’.

#### 3.5.3.5精确查询

## 3.6界面美化

### 3.6.1程序启动界面

在MainWindow.cpp中，利用QTime让程序等待一段时间，并使用QElapsedTimer不停地处理事件，以使得程序保持响应。

//程序启动界面延时

QElapsedTimer t;

t.start();

while(t.elapsed()<1000)

{

QCoreApplication::processEvents();

}

在main.cpp中利用QSplashScreen显示启动画面，QPixmap载入图片：

//加载并显示启动画面

QPixmap pixmap(":/res/start");

QSplashScreen splash(pixmap);

splash.setDisabled(true);

splash.show();

在程序启动画面显示完后调用主界面，结束程序启动界面

MainWindow w;

splash.finish(&w);

### 3.6.2插入图片

直接插入图片并不会根据窗口widget或文本框label大小自动调整，需要调整到所需的大小，需要让label中的图片根据widget的大小重新resize：

//插入图片（图片随widget大小而变）

QPixmap img(":/res/shopping");

ui->label->setPixmap(img);

ui->label->setScaledContents(true);

ui->label->resize(ui->widget->size());//调整大小

ui->label->show ();：

### 3.6.3修改图标

用在线网页转换图片为.ico格式放在项目文件夹中，然后在项目文件夹中新建文本文件，输入以下代码：

IDI\_ICON1 ICON DISCARDABLE "icon.ico"

其中icon.ico为图片的名称

将文本文件重命名为myicon.rc

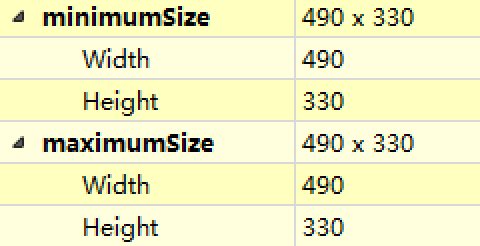
并在项目pro文件中添加：RC\_FILE = myicon.rc

重新编译即可修改图标如图：

### 3.6.4界面大小的固定

不同于代码编写的界面，直接可以用 this->setFixedSize( this->width (),this->height ());固定

对于QtDesigner编写的界面，将界面的最大值和最小值调整一致即可。



## 3.7关于容错

### 3.7.1消息框的应用

防止用户输入导致程序出错，在输入的环节设置了语句判断，并用MessageBox进行提醒用户输入。

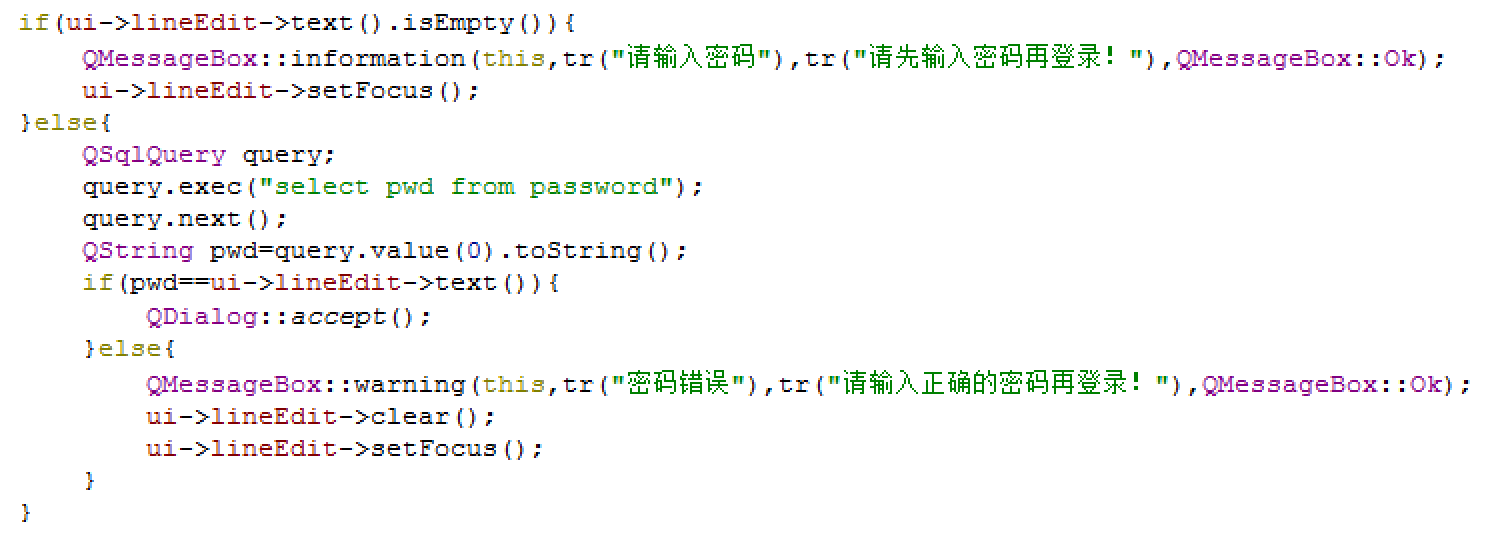
例如密码输入环节，若用户没有输入密码：



若输入错误密码：



其实现是在“登录”按钮的槽函数下添加判断是否输入和判断密码是否正确：

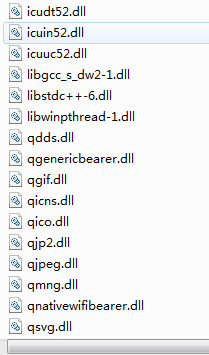


利用QMessageBox类，修改其information的提示内容，从而达到对于输入的提醒，起到良好的容错处理。

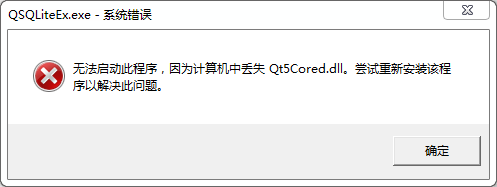
# 4项目总结

## 4.1Qt静态编译的实现

由于Qt是默认动态编译的，因此编译出的可执行文件往往需要携带大量动态库文件如下图：



否则就会出现类如下图警示：



在正式编写之前，需先解决静态编译的问题。

这里参考的是一位博主的实现方式：

<http://www.cnblogs.com/elect-fans/archive/2012/03/24/2415737.html>

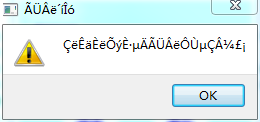
且鉴于Qt5和Qt4的版本差异较大，例如在Qt5的安装文件中无法找到Command Prompt软件实现批处理。考虑到Qt4和Qt5的功能使用方面差异不大，最终使用Qt4.8.4实现静态编译，双击可执行文件（.exe）即可完美运行。

## 4.2中文乱码的解决

鉴于采用的是Qt4.8.4全英文编写软件，不同于Qt5中已实现汉化，对于汉字的兼容性并不良好。

在Qt Designer的ui中直接输入的中文可以正常显示，然而对于代码中使用QString输入的中文会显示乱码。

例如QMessageBox中的乱码：



对此一些网上资料中解释为没有采用UTF-8或GBK的编码，然而加上形如

QTextCodec::setCodecForLocale(QTextCodec::codecForName("GBK"));

QTextCodec::setCodecForLocale(QTextCodec::codecForName("UTF-8"));代码仍然没有作用

这个问题在网上搜了很久，最终看到另一位博主遇到相同问题的解决方案

<http://blog.csdn.net/mociml/article/details/5531643>

原因发现是字库问题，QMessageBox默认的字库不支持中文，在主调函数中添加如下四行代码即可完美解决：

QTextCodec::setCodecForCStrings(QTextCodec::codecForLocale());

QTextCodec::setCodecForTr(QTextCodec::codecForLocale());

QFont font("宋体",9,QFont::Bold,FALSE);

a.setFont(font);

## 4.3数据库中文乱码的问题

延续之前碰到的中文乱码显示问题，在数据库建立和读取数据库的过程也碰到了乱码问题，究其原因还是由于Qt4不支持中文版，所以需要在插入数据时先行编码，再手动用代码设置为本地编码读取。否则会出现形如下图：



插入数据时需要先修改编码，故不同于一般的插入操作：query.exec(QString(tr("insert into…

需改为query.exec(QObject::tr("insert into…

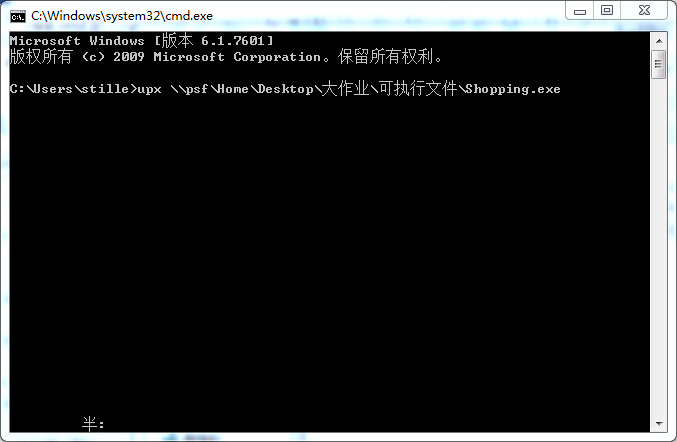
考虑到数据库的读取和建立在connection.h文件中，而在主调函数中的修改编码不起作用，因此需要在connection头文件中加入以下代码：

QTextCodec::setCodecForTr(QTextCodec::codecForLocale());即可正常显示。

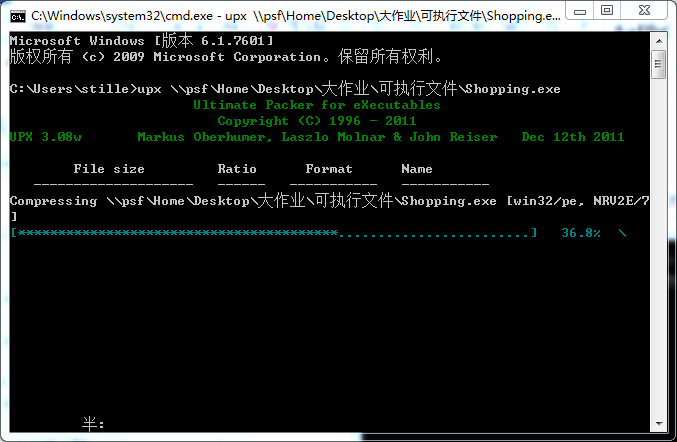
## 4.4关于压缩

一般静态编译生成的可执行程序都较大，可以将可执行文件压缩

使用upx压缩生成的可执行文件，安装好后，在cmd中输入upx 路径



压缩比例一般在30%到40%还是相当可观的。



压缩前

C:\Users\stille\AppData\Roaming\Tencent\Users\948600671\QQ\WinTemp\RichOle\{{S9Y5Q_O2T%@V3I0]%`L_F.png

压缩后

C:\Users\stille\AppData\Roaming\Tencent\Users\948600671\QQ\WinTemp\RichOle\A`K}]QO6N@%T11_{NJZ@QJX.png

## 4.5关于相关资料的搜寻和自学

在开始此次小学期以前并没有接触过可视化界面的编写以及Qt的使用，因此对于很多类和函数的使用需要在实践中摸索，虽然最终都实现了这些功能，但还记得起初连最简单的窗口连接的那两行代码：

void MainWindow::on\_loginButton\_clicked()

{

Login \*ui=new Login;//建立

ui->show();//显示

}

都研究了好几天，因为书本和网上大多数都是直接编写界面，即直接new一个界面。

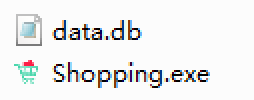
此外对于插入合适的图片大小的四行代码、修改中文乱码、静态编译的实现，每一个问题的解决都是搜寻了许多博客、网页，参考网友遇到相似的问题，进行一次次修改和一次次尝试，最终解决的，至今还能记得刚刚静态编译成功时双击.exe文件再也不会提示缺失.dll脚本文件时候的激动。在这个过程中发现了很多很好的论坛网站例如CSDN，CNBlog等。

也在实践中发现，许多函数的运用并不是直接查询了该类或该函数的形参就能直接使用的，往往还要调试，逻辑判断哪里出了问题。

例如解决数据库中文乱码时调试了好几天，最后修改了数据库建立的语句全为英文，于是想到可能数据库并没有被重写，后来弄明白，由于设置了primary key，因此重新insert的语句并没有起到作用，删除了原先的data.db，重新编译，总算成功了。

# 5相关问题的说明

由于程序的数据都储存在data.db文件，需要将Shopping.exe文件和数据库文件放在一个文件夹下，双击即可完美运行：



此外，程序的初始密码为“123456”.