目录

[设计理念及使用说明 2](#_Toc519937631)

[**1.** **登录窗口：** 2](#_Toc519937632)

[**2.** **主窗口：** 3](#_Toc519937633)

[**3.** **查看学分绩：** 4](#_Toc519937634)

[**4.** **详细成绩：** 4](#_Toc519937635)

[**5.** **查看GPA：** 5](#_Toc519937636)

[**6.** **天气预报：** 7](#_Toc519937637)

[实现思路 7](#_Toc519937638)

[**1.** **学号及密码的输入、保存和读取：** 7](#_Toc519937639)

[**2.** **发送post请求** 8](#_Toc519937640)

[**3.** **获取页面数据：** 11](#_Toc519937641)

[**4.** **处理页面数据：** 11](#_Toc519937642)

[**5.** **登录错误情况：** 13](#_Toc519937643)

[**6.** **实现分页操作：** 13](#_Toc519937644)

[**7.** **调用api接口获取天气信息：** 14](#_Toc519937645)

[遇到的问题 14](#_Toc519937646)

[**1.** **无法获取信息：** 15](#_Toc519937647)

[**2.** **解码返回信息：** 15](#_Toc519937648)

*github账号：chendante*

*github项目地址：https://github.com/chendante/NKer-Qt.git*

# Qt大作业——南开人考试信息获取助手

## 设计理念及使用说明

1. **登录窗口：**



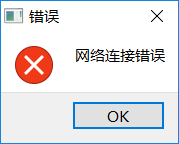
运行程序后会直接进入该窗口，让用户进行登录，用户将保存按钮选中后，会将用户的学号密码信息进行保存，下次用户进入该窗口后，会直接将保存的学号密码信息读出在输入框中。

登录的学号密码应与南开大学教学管理信息系统（<http://eamis.nankai.edu.cn>）相一致。

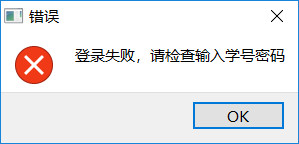
登录的学号只能输入数字（因为南开大学所设置的学号只有数字），登录密码可以有数字字母或者符号，但是不允许中间出现空格（与密码应有格式相一致）。

如果用户没有登录便将该对话框关闭，便会将整个程序关闭。

如果用户网络连接（比如没有网络连接）有问题，或者南开大学教学管理信息系统的服务器出现问题（如在选课高峰期），会跳出错误提示框：



而当用户输入的学号密码不正确时，也会跳出错误提示框：



以上两个提示框关闭后，会回到登录界面。

1. **主窗口：**

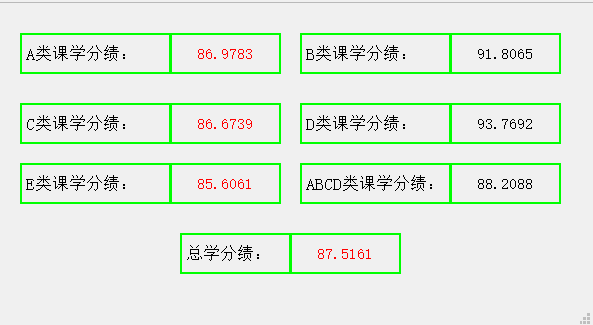


登录成功后将会进入主窗口，标题会变成“欢迎：登录人姓名 学号：登录人学号“的样式。

主窗口一共有三个功能，分别是：“查看学分绩“、”详细成绩“、”查看GPA”，分别绑定的快捷键为：“CTRL+A”、“CTRL+B”、“CTRL+C”。

刚进入主窗口会直接进入查看学分绩页面，根据所触发菜单项或工具栏功能进行页面的切换。

1. **查看学分绩：**



该界面为一进入主界面就会显示的页面。当跳转到其它页面后，也可以通过菜单栏功能下的查看学分绩或者工具栏下的查看学分绩按钮，或者直接通过CTRL+A快捷键跳转回该页面。

该显示了计算得到的各类课的学分绩，和我们常用的ABCD类课学分绩和总学分绩。

我们可以看到，有的学分绩显示颜色为红色，有的则是黑色。各类课程学分绩的颜色由总学分绩来决定，比总学分绩高的显示为黑色，而比总学分绩低的显示为红色，提示用户应该注意这类课程。而总学分绩的颜色，当小于90时，将会置为红色，当大于等于90时，将会变为黑色，提示用户应当继续努力。

ABCDE类课的分类来自南开大学教学管理信息系统的计划完成情况页面。A类课代表：校公共必修课；B类课代表：院系公共必修课；C类课代表：专业必修课；D类课代表：专业选修课；E类课代表任选课。

1. **详细成绩：**

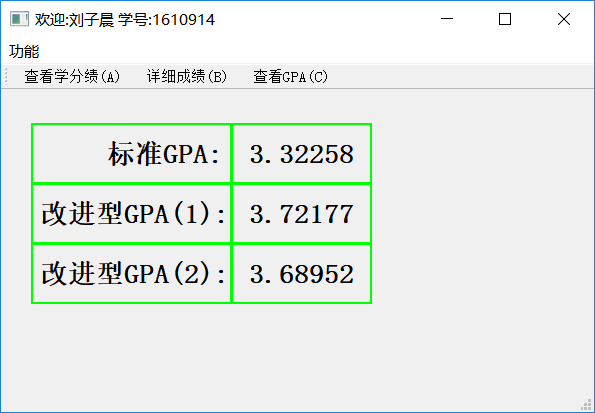


点击查看详细成绩后会跳转至该页面，在该页面用户可以查看已经出了的所有成绩（即在 南开大学教学管理信息系统的计划完成情况页面 已经显示的成绩。所给出的信息分别有：课程类型、课程名称、成绩、学分。

用户可以在该界面查看各个课程的详细信息，得知自己在每门课上的得失。

通过上图，我们可以看到有的成绩被标为红色，有的成绩被标为绿色，有的成绩为黑色。红色代表该门成绩低于85分，提示用户注意该项成绩；绿色代表该门成绩高于90分，提示用户该门课程掌握的不错；其它在两个分中间的则显示为默认的黑色。学分当大于3分（不包括3分）且不是E类课时将被标为蓝色，提示用户注意该项课程，它所占比重很高。

1. **查看GPA：**



点击 查看GPA 后会跳转至该页面，在该页面将会显示通过计算所有成绩所得出的GPA。GPA的计算方式有很多种，这里显示常用的三种GPA计算方式：

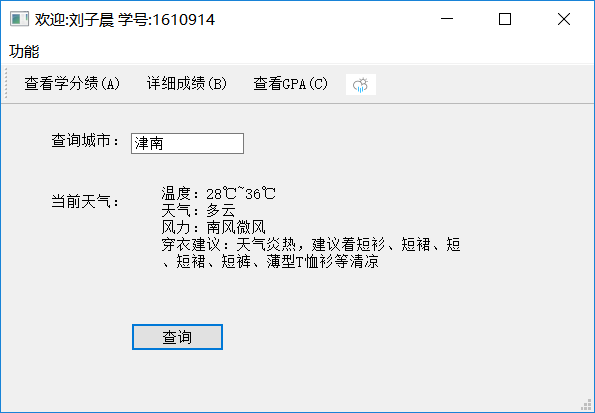
|  |  |
| --- | --- |
| **(1)标准4.0** | |
| 成绩 | GPA |
| 100～90 | 4.0 |
| 89～80 | 3.0 |
| 79～70 | 2.0 |
| 69～60 | 1.0 |
| 59～0 | 0 |

|  |  |
| --- | --- |
| **(2)改进4.0(1)** | |
| 成绩 | GPA |
| 100～85 | 4.0 |
| 84～70 | 3.0 |
| 69～60 | 2.0 |
| 59～0 | 0 |

|  |  |
| --- | --- |
| **(3)改进4.0(2)** | |
| 成绩 | GPA |
| 100～85 | 4.0 |
| 84～75 | 3.0 |
| 74～60 | 2.0 |
| 59～0 | 0 |

以上三个GPA计算方式分别对应显示的三种GPA。

1. **天气预报：**



点击工具栏的天气图标可以进入本页面，刚进入时默认查询城市为津南，可以进行更改，选好查询城市后点击查询，可以得到当前的天气信息，包括温度、天气、风力、穿衣建议。

## 实现思路

1. **学号及密码的输入、保存和读取：**

* **输入：**

首先需要将密码框的echoMode设为password，这样可以轻松实现，密码以圆点格式显示，并不能被复制粘贴，可以增加软件的安全性。

而且，在设计过程中考虑到学号及密码的格式问题，应当设计学号只能输入数字，而密码中间不能有空格，这样可以方便用户纠错，而且符合常理。该项功能的实现借助了正则表达式和QValidator类及下面的函数：

**void** QLineEdit::**setValidator**(const [**QValidator**](../qtgui/qvalidator.html) \**v*)

具体实现代码：

QRegExp regx("[0-9]\*");//正则表达式,学号的输入只能有数字

QValidator \*validator = new QRegExpValidator(regx, this );

ui->lineEdit->setValidator( validator );//设置学号的输入验证

QRegExp regxPW("^[\\S]\*$");//设置不允许密码中间输入空格

QValidator \*pwvalidator = new QRegExpValidator(regxPW, this );

ui->lineEdit\_2->setValidator( pwvalidator );//设置密码验证器

* **保存：**

在登录界面有一个checkbox，当其被选中时，在点击登录按钮后，会将两个输入框中的内容保存在根目录下的"/login.txt"文件中，该功能通过QFile类及QTextStream类来实现：

**bool** QFile::**open**(**[QIODevice::OpenMode](qiodevice.html" \l "OpenModeFlag-enum)** *mode*)

*mode* 有3种选择 [QIODevice::ReadOnly](qiodevice.html#OpenModeFlag-enum)（只读）、 [QIODevice::WriteOnly](qiodevice.html#OpenModeFlag-enum)（只写）、[QIODevice::ReadWrite](qiodevice.html" \l "OpenModeFlag-enum)（可读可写）。

具体代码实现：

//当保存按钮被选中后，将账号密码保存在txt中

if(ui->checkBox->isChecked())

{

QFile f1("/login.txt");

//将文件以只写形式打开

f1.*open*(QIODevice::WriteOnly|QIODevice::Text);

QTextStream str(&f1);

str<<id<<endl<<password;

f1.*close*();

}

* **读取：**

当程序运行时，将根目录下的"/login.txt"文件打开，将第一行的数据传入学号，第二行的数据传入密码。功能实现方法跟保存类似：

具体代码实现：

//读取已经保存的账号密码信息，并将其置于登录框中

QFile file("/login.txt");

file.*open*(QIODevice::ReadOnly| QIODevice::Text);

QTextStream txtInput(&file);

QString str1=txtInput.readLine();

ui->lineEdit->setText(str1);

QString str2=txtInput.readLine();

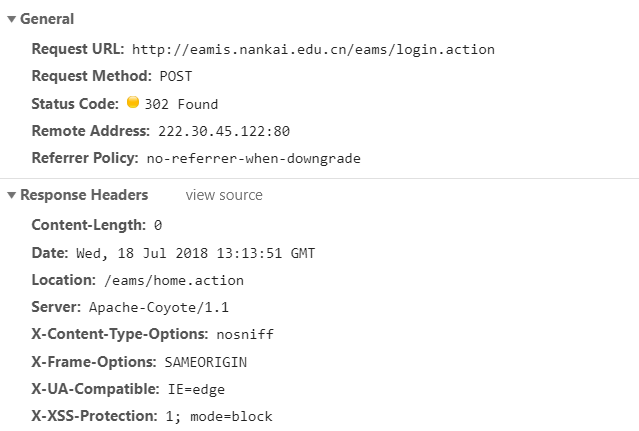
ui->lineEdit\_2->setText(str2);

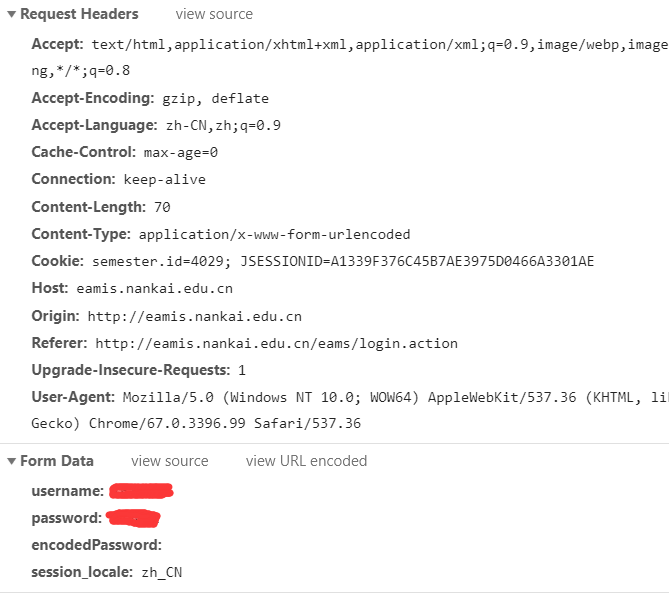
file.*close*();

1. **发送post请求**

当用户点击登录后，我们需要根据用户输入的学号密码登录南开大学教学管理系统，以获取信息，这需要用到Qt Network模块中的QNetworkRequest、QNetworkReply和QNetworkAccessManager几个类来使用通用的协议来执行网络操作。

我们先从南开大学教学管理系统的登录界面进行一次登录操作，得到登录的post请求信息：





D://CloudMusic/NoteYoudao/danteliuzichen@163.com/21e451d7a40445c6ab666e2b5f72eead/clipboard.png

（图中遮住的部分为作者的学号或者密码）

通过得到的信息，设置post请求的参数：

具体实现代码：

//设置 request headers

QNetworkRequest m\_req(QUrl("http://eamis.nankai.edu.cn/eams/login.action")); //该网址为教学管理系统登陆网址

m\_req.setHeader(QNetworkRequest::ContentTypeHeader,"application/x-www-form-urlencoded");

//设置 Form Data

QString content=tr("username=%1&password=%2&encodedPassword=&session\_locale=zh\_CN").arg(id).arg(password);

m\_login->post(m\_req,content.toUtf8());

这样，我们就向该网站的服务器发送了一个登录的post请求。

收到请求后，服务器会向我们客户端发回一个应答，QNetworkAccessManager类除了发送请求还可以接收应答，我们将管理器的finished()信号与相应处理这一信号的自定义槽函数相关联：

QObject::connect(m\_login, SIGNAL(finished(QNetworkReply\*)), this, SLOT(finishedLogin(QNetworkReply\*)));

1. **获取页面数据：**

在得到登录的post请求的返回信号后，我们用发送post请求的类似方式发送一个get请求（注意两个请求发送的 QNetworkAccessManager类 的对象需要是同一个）：

//访问计划完成情况页面

QNetworkRequest m\_req(QUrl("http://eamis.nankai.edu.cn/eams/myPlanCompl.action"));

m\_req.setHeader(QNetworkRequest::ContentTypeHeader,"application/x-www-form-urlencoded");

m\_login->get(m\_req);

get请求同样会使服务器向我们的客户端发回一个应答，我们通过关联槽函数来处理返回的数据。这一get请求会使我们得到计划完成情况页面的所有信息，该页面的编码格式为utf-8，我们通过**[Q](qbytearray.html)ByteArray** QIODevice::**readAll**()以及QTextCodec类将其转码为QString格式：

QTextCodec \*codec = QTextCodec::codecForName("UTF-8");

QString score\_data= codec->toUnicode(m\_reply->readAll());

1. **处理页面数据：**

我们通过get请求得到的页面数据先在loginDialog中获取到，而后通过信号与mainWindow进行绑定，将得到的数据传送到mainWindow中，在这一模块将数据进行处理：

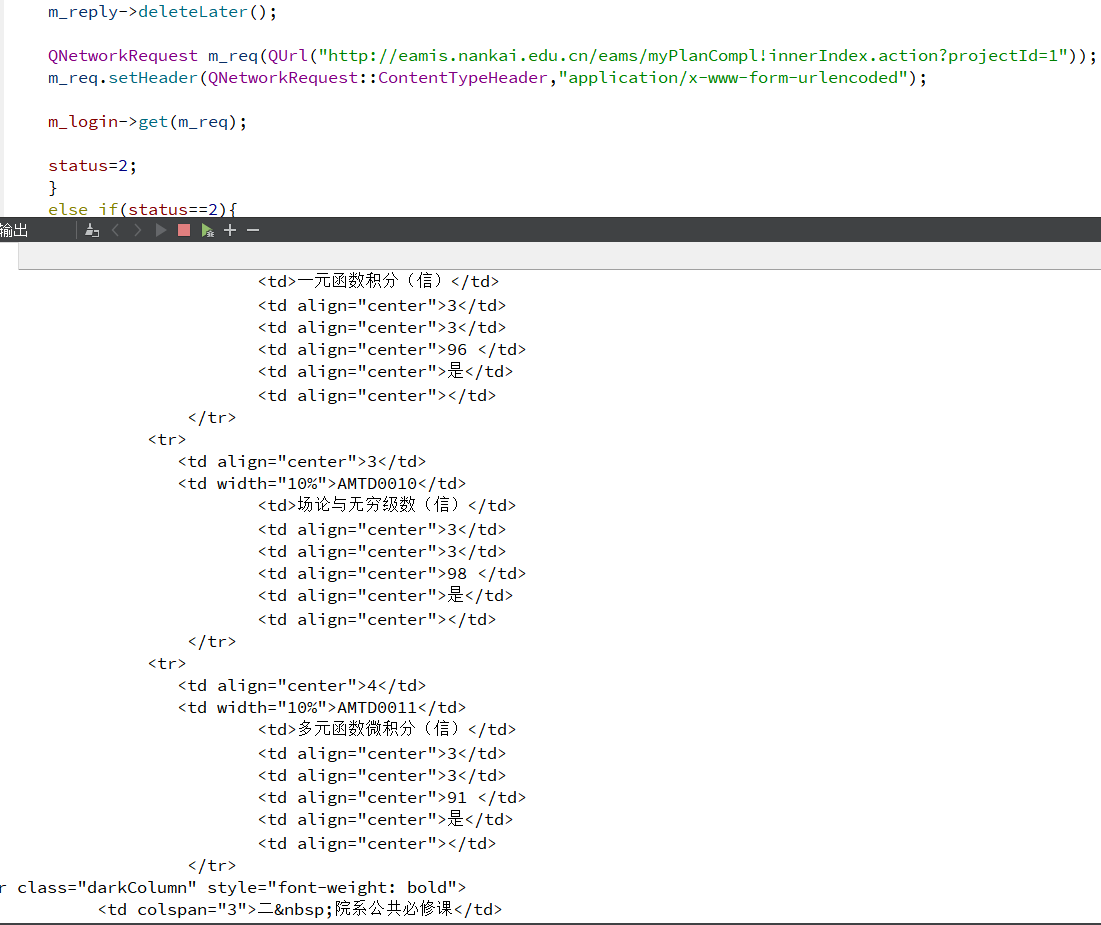
发送信号：

emit getScore(score\_data);

绑定：

a.connect(&dlg,LoginDialog::getScore,&w,MainWindow::dealScore);

得到的页面数据在经过转码后为：



我们可以看到，我们得到的数据格式是html代码的形式，我们需要将这些数据处理为我们所需要的格式。由于这是一长串字符串，我们需要从中得到想要的信息，所以我们采用正则表达式的形式来获取数据：

//正则表达式，用于选取页面中的有用信息

QString rul\_id=(">""([0123456789]{7})""<");//学号

QString rul\_name=("18%\">""([^2016:]{2,20})""</td>");//姓名

QString rul\_socre=("</td>\r\n\t\t\t \t<td>([^烫]{1,50})</td>\r\n\t\t\t \t<td align=\"center\">([0123456789. ]{0,10})</td>\r\n\t\t\t \t<td align=\"center\">([0123456789. ]{0,10})</td>\r\n\t\t\t \t<td align=\"center\">([0123456789. ]{0,10}) </td>\r\n\t\t\t");

QString rul\_type1=("院系公共必修课");

QString rul\_type2=("专业必修课");

QString rul\_type3=("专业选修课");

QString rul\_type4=("任选课");

这些正则表达式，是通过对得到的数据格式进行分析得出的。

下面给出一个具体例子，展示如何使用正则表达式获取相应数据：

QRegExp rx(rul\_name);

int pos1 =score\_data.indexOf(rx);

if( pos1!=-1){name=rx.cap(1);}

而后将得到的数据进行处理，可以得到用户所有已经修完的课程的名称、类型、成绩、学分信息，通过计算，可以得到用户的各类成绩的学分绩和GPA。

1. **登录错误情况：**

登录并不是总会成功，登录不成功的情况可以大致分为两种，一是：网络连接有问题，找不到服务器，或者服务器出现问题、二是：输入的学号密码有问题。对于这两种情况，我们采用不同的方式进行判断。

一、网络连接有问题：这种情况，返回的QNetworkReply\*中有一个私有成员变量可以判断，只需要用error()函数将其调用即可，该变量只有在没有错误情况出现时才会为零。

二、学号密码有问题：这种情况下，发回的应答并不会显示错误，而当我们进行下一步的get请求时却不会有返回值出现，所以只需要再进行下一步时，判断返回数据是否为空即可。

1. **实现分页操作：**

对于每一个不同的页面，由于显示的信息我们需要展示在主窗口的相同位置，所以这就需要我们实现分页操作，这一操作需要依赖QStackedWidget类来进行实现。我们首先创建好有各个分页需要显示的内容的对话框，而后将其添加进该类的对象中，并将该变量的显示位置设定好，每当触发换页操作时，将对应显示页面的编号设定好即可实现该操作：

TableDialog \*tableDlg=new TableDialog(this);

GPADialog \*gpaDlg=new GPADialog(this);

WeatherDialog \*weatherDlg=new WeatherDialog(this);

m\_stackedWidget->addWidget(tableDlg);

m\_stackedWidget->addWidget(gpaDlg);

m\_stackedWidget->addWidget(weatherDlg);

m\_stackedWidget->move(0,35);

m\_stackedWidget->resize(600,600);

m\_stackedWidget->hide();

触发换页操作：

int index=1;

m\_stackedWidget->show();

m\_stackedWidget->setCurrentIndex(index);

1. **调用api接口获取天气信息：**

首先我们需要找到一个合适的api接口，并申请获得使用的密钥，从而获取到使用权限。这里我们使用的是 *聚合数据api* 中的 *全国天气预报api接口。*

通过api接口获取信息即为向相应的url发送get请求，与之前我们进行过的get请求类似，服务器同样会发回一个应答，但是与之前的应答不同，该应答的数据格式为Json格式，这需要我们调用QJsonObject类对其进行处理，从而获得我们所需要的信息，并将其转码为QStirng类型，从而可以显示在label上：

QJsonObject data=QJsonDocument::fromJson(m\_reply->readAll()).object();

QJsonObject today=data.value("result").toObject().value("today").toObject();

QString temperature=today.value("temperature").toString();

api接口的使用文档：



通过该文档的信息，我们设置发送get请求的各项参数。

## 遇到的问题

在实现过程中，我遇到过很多的难题，这里挑几个对我印象深刻的说一说。

1. **无法获取信息：**

在进行网络操作过程中，由于对于一个QNetworkAccessManager类的变量的finished信号是相同的，所以对于发送不同的请求，我倾向于使用不同的QNetworkAccessManager变量，比如登录用一个变量，获取计划完成页面数据再使用另一个变量。但是在实际操作过程中却出现了问题，当我登录完之后，再用另一个变量进行获取计划完成页面数据时却出现了问题，当时我还不知道这种返回效果与未登录一模一样，再经过多次实验、查询和猜想后，我知道这种情况的出现可能与网络协议有关，就如同，我们用一个浏览器登录后，当我们用另一个浏览器时，还需要再进行一次登录操作。

1. **解码返回信息：**

从网页获取到的信息也是不尽相同，不同的请求获取到的信息的编码格式也不同，这就需要我们对于各种返回信息的编码格式先进行一个大致判断再进行多种尝试，最终确定其编码格式，再进行解码，然后处理成我们需要的信息。

在进行正则表达式的设置过程中，我也遇到了很多问题，也是经过猜想和多种尝试进行解决的。