东软睿道

项目实训总结报告

项 目 名 称: ColorMail简易邮箱系统

学 校 名 称: 北京理工大学

项 目 讲 师: 卢春江

姓 名: 薛佳伟

学 号: 1120183494

实 训 日 期: 2020年9月7日-9月17日

**1 引言**

1.1项目概述

本项目是以Qt为平台，利用QTcpSocket与QTcpServer来实现客户端和服务端之间的连接，以此来做一个简易的校园网内邮件系统。本系统利用了许多Qt中类似于QTcpSocket、QJsonArray等打包好了的包，给我们的开发提供了很大的方便。

1.2 需求分析

本项目的邮件系统具有很多实用功能，包括联系人的建立、附件的实时传送，可以为人们实时传输信息提供极大的方便。

1.3 运行环境

i5-7200U,MX130

Ubuntu 20.04.1LTS 64bit

**2 项目设计**

2.1 设计思路

在项目中，我负责信号连接方面的功能。我利用了Qt打包好的QTcpSocket和QTcpServer来进行信号传输。将传输的信号的类型分成普通型和附件型，在接收端做好标记，分别用不同的方式进行接收。

2.2 模块功能介绍

信号流传输部分一共分为两个模块，一个是针对普通信号的传送，一个是针对附件的传送。普通信号可以传送邮件的发送者、接受者、主题、时间等内容，附件信号可以传送用户或者服务器需要传输的附件信号流，并在服务器或者客户端得到该数据流，得到完整的附件信息。

2.3 模块结构图

客户端1 QByteArray 服务器 QByteArray 客户端2

QTcpSocket QTcpSocket

2.4 程序流程图

（1）普通信号的传输：

QString、int… ——> QJsonArray ——> QJsonDocument

——> QByteArray

（2）附件的传输：

qint64-QTcpSocket->write(QByteArray)

2.5 功能设计分工

针对信号传输的两个模块，在服务端和客户端中由我来负责QTcpSocket的传输和QTcpServer的接收。

**3 详细设计**

若客户端给服务器发送的信息为字符串或数字类型等，则将这些信息保存在QJsonDocument中，以QByteArray的形式由QTcpSocket->write传输至服务端，在服务端中，以QByteArray的形式读出再转回QJsonDocument；若客户端发送的是附件，则以QDataStream的形式转QByteArray发送和接收。在发送信号流的时候做出标记，以便识别出这是一组什么样的信号流，需要实现什么样的功能。由服务端传输信号回客户端也是如此。

**4 实现方案**

1.普通信号的传输，可以将邮件的基本信息（发送者和接受者、主题、内容）发送至服务端或者从服务端发送回客户端：

a.发送至服务端：

QString currenttext = ui->textEdit->toPlainText(); //邮件内容

QString currentsubject = ui->lineEdit\_subject->text(); //邮件主题

QString currentreceiver = ui->lineEdit\_receiver->text(); //邮件发送者

QString currenttext = ui->textEdit->toPlainText(); //邮件内容

QString currentsubject = ui->lineEdit\_subject->text(); //邮件主题

QString currentreceiver = ui->lineEdit\_receiver->text(); //邮件发送者

QJsonArray txt\_arr;

QDateTime time = QDateTime::currentDateTime();

QString currenttime = time.toString("yyyy-MM-dd hh:mm:ss");

txt\_arr.append(4);//此处标记信号类型

txt\_arr.append(name);//发送者用户名

txt\_arr.append(currentreceiver);

txt\_arr.append(currentsubject);

txt\_arr.append(currenttext);

txt\_arr.append(currenttime);//时间信息

txt\_arr.append(Draft\_flag);

txt\_arr.append(file\_flag);

QJsonDocument txt\_doc;//转为QJsonDocument类型

txt\_doc.setArray(txt\_arr);

QByteArray txt\_bt = txt\_doc.toJson(QJsonDocument::Compact);//转为QByteArray类型

tcpClient->write(txt\_bt);//发送信号

b.服务端的接收：

void Server::**startserver**()

{

if(!m\_tcpServer->listen(QHostAddress::Any, 6666))

{

qDebug() << "connect error!";

}

else

{

bytesReceived = 0;

totalBytes = 0;

filenameSize = 0;

connect(m\_tcpServer, SIGNAL(newConnection()), this, SLOT(newConnect()));

}

}

void Server::**newConnect**()

{

bytesWritten = 0;

perDataSize = 64 \* 1024;

m\_tcpSocket = m\_tcpServer->*nextPendingConnection*();

connect(m\_tcpSocket, SIGNAL(readyRead()), this, SLOT(Read()));

}

2.附件信息的传输，可以将附件的信号传输到服务端或客户端：

a.发送至服务端：

void sentEmail::**Transferfile**() //传送文件大小，名称等内容

{

tcpFile->*connectToHost*("192.168.43.188", 6666);//连接服务器

localFile=new QFile(fileName);

if(!localFile->*open*(QFile::ReadOnly))

{

ui->label\_notice->setText("文件错误！");

qDebug()<<"client：open file error!";

return;

}

totalBytes=localFile->*size*(); //文件数据大小

QDataStream sendOut(&outBlock,QIODevice::WriteOnly);

sendOut.setVersion(QDataStream::Qt\_5\_14); //序列化格式版本号

QString currentFileName=fileName.right(fileName.size()-fileName.lastIndexOf('/')-1); //文件总大小、文件名大小、文件名

sendOut<<qint64(0)<<qint64(0)<<currentFileName; //发送附件

totalBytes+=outBlock.size(); //文件数据+文件信息大小

sendOut.device()->*seek*(0);

sendOut<<totalBytes<<qint64(outBlock.size()-sizeof(qint64)\*2); //返回outBolock的开始，用实际的大小信息代替两个qint64(0)空间

bytesToWrite=totalBytes-tcpFile->write(outBlock); //发送完数据后剩余数据的大小

outBlock.resize(0); //清空缓存区

}

void sentEmail::**updateClientProgress**(qint64 numBytes)

{

qDebug() << "aha";

bytesWritten+=(int)numBytes; //已经发送数据的大小

if(bytesToWrite>0)

{

outBlock=localFile->read(qMin(bytesToWrite,payloadSize)); //每次发送loadSize大小的数据，这里设置为64KB，如果剩余的数据不足64KB就发送剩余数据的大小

bytesToWrite-=(int)tcpFile->write(outBlock);//发送完一次数据后还剩余数据的大小

outBlock.resize(0); //清空发送缓冲区

}

else

{

localFile->*close*(); //如果没有发送任何数据，则关闭文件

}

if(bytesWritten==totalBytes) //发送完毕

{

localFile->*close*();

tcpFile->*close*();

}

}

b.客户端接收附件：

QDataStream inFile(this->m\_tcpSocket);

inFile.setVersion(QDataStream::Qt\_5\_15);

if (bytesReceived <= sizeof(qint64)\* 2)

{

if ((m\_tcpSocket->*bytesAvailable*() >= (sizeof(qint64)) \* 2) && (filenameSize == 0))

{

inFile >> totalBytes >> filenameSize;

bytesReceived += sizeof(qint64)\* 2;

}

if ((m\_tcpSocket->*bytesAvailable*() >= filenameSize) && (filenameSize != 0))

{

inFile >> filename;

bytesReceived += filenameSize;

filename = "E:/QT\_Document/ServerApp/database/"+filename;

//qDebug() << filename;

QString sql7;

QSqlQueryModel \*model7=new QSqlQueryModel;

sql7="select count (\*) from Mail where send\_time='"+s\_times+"'";

//qDebug() << s\_times;

model7->setQuery(sql7);

QModelIndex index7=model7->*index*(0,0);

if(index7.data()!=0)

{

QString sql6;

sql6="update Mail set address = '"+filename+"' where send\_time='"+s\_times+"'";

QSqlQueryModel \*model6=new QSqlQueryModel;

model6->setQuery(sql6);

}

localFile = new QFile(filename);

if (!localFile->*open*(QFile::WriteOnly))

{

qDebug() << "Server::open file error!";

return;

}

}

else

return;

}

if (bytesReceived < totalBytes)

{

bytesReceived += m\_tcpSocket->*bytesAvailable*();

inBlock = m\_tcpSocket->readAll();

localFile->write(inBlock);

inBlock.resize(0);

}

if (bytesReceived == totalBytes)

{

bytesReceived = 0;

totalBytes = 0;

filenameSize = 0;

File\_Flag = 0;

localFile->*close*();

}

**5 系统测试**

针对信号问题，我将同学的前后端代码同时deBug。因为每个信号的传输都有特殊的标记，因此我在服务端deBug的时候可以及时地看到每次传输过来的信号是否是正确的，从而从中找取Bug。用这种方法可以较快地找出是前端还是后端（数据库）的问题。

在附件传输的过程中有一个比较细的关于信号的bug。即在每次传输附件时，是通过每次读取到附件后先发送一个特定的标记到接收端，然后用connect函数启动接收函数，由于对Qt环境的不熟悉，前端同学将启动函数在头文件中放到了普通函数中。这个困扰了我们几个小时的问题最后在比对中找出，从而将启动函数放入【slots：】中。

**6 心得体会**

在本次项目实训中，我们小组四位同学的基础都很一般，对Qt操作和Linux系统都不是很熟悉，更不要说实际的编程和写软件了。因此在一周左右的高强度开发中，我们彼此配合，每天都一起商量好安排接下来的学习内容和任务。这是让我体会的最深的一点，即同组同学之间的配合与默契在合作开发中特别重要。

其次，我负责的是前后端的信号连接，在deBug的时候同时需要找前后端两边的问题，因此也让我对deBug的过程更加印象深刻，对断点以及实时显示数据值的利用更加熟练。

很感谢这一次的专业实习，让我第一次真正上手写一个可以真正使用的软件，让我们小组成员有了巨大的成就感，也让我们真正地体会到了做一个合格的程序员的不简单。这一次印象深刻的实习让我体会到了编程的快乐和默契配合的爽快，也让我坚定信心将这一门学科学好，为当一个合格的程序员做好准备！