编号：



计算机网络课程设计(论文)说明书

题 目：**基于UDP协议网上聊天程序**

系 别： 计算机与信息安全

专 业： 软件工程

学生姓名： 薛天杭

学 号：  1400330129

指导教师： 王虎寅

2019年 04月15日

目 录

[1 设计任务及目标 1](#_Toc8351558)

[1.1 设计任务 1](#_Toc8351559)

[1.2任务要求 1](#_Toc8351560)

[1.3设计目标 1](#_Toc8351561)

[2 需求分析 2](#_Toc8351562)

[2.1 用户及业务需求 2](#_Toc8351563)

[2.2 层次方框图 3](#_Toc8351564)

[2.2.1用户层次方框图 3](#_Toc8351565)

[2.2.2管理员层次方框图 4](#_Toc8351566)

[2.3 数据流图 5](#_Toc8351567)

[2.3.1注册数据流图 5](#_Toc8351568)

[2.3.2登录数据流图 5](#_Toc8351569)

[2.3.3聊天数据流图 6](#_Toc8351570)

[3 整体设计 7](#_Toc8351571)

[3.1 客户端设计 7](#_Toc8351572)

[3.1.1登录页面设计 7](#_Toc8351573)

[3.1.2注册页面设计 8](#_Toc8351574)

[3.1.3聊天页面设计 9](#_Toc8351575)

[3.2 服务端设计 10](#_Toc8351576)

[3.3程序流程 11](#_Toc8351577)

[4 系统实现 12](#_Toc8351578)

[4.1 服务端的实现 12](#_Toc8351579)

[4.1.1登录功能的实现 12](#_Toc8351580)

[4.1.2注册功能的实现 14](#_Toc8351581)

[4.1.3聊天功能的实现 15](#_Toc8351582)

[4.2 客户端的实现 16](#_Toc8351583)

[4.2.1登录页面 16](#_Toc8351584)

[4.2.2注册页面 17](#_Toc8351585)

[4.2.3聊天页面 19](#_Toc8351586)

[5 软件测试 21](#_Toc8351587)

[5.1 登录功能 21](#_Toc8351588)

[5.2 注册功能 21](#_Toc8351589)

[5.3 聊天功能 22](#_Toc8351590)

[6 总结 23](#_Toc8351591)

[6.1 已完成的功能模块 23](#_Toc8351592)

[6. 2 未完成的功能模块 23](#_Toc8351593)

[6. 3个人小结 23](#_Toc8351594)

[6.2 展望 23](#_Toc8351595)

[参考文献 24](#_Toc8351596)

# 1 设计任务及目标

## 设计任务

设计一个基于UDP协议网上聊天程序

## 1.2任务要求

实现一简单的聊天程序实现网上聊天，包括服务器和客户端。要求：

（1）支持多人聊天。

（2）客户端具有图形化用户界面。

## 1.3设计目标

完成一款拥有图形化界面的聊天软件的设计，能够满局域网内聊天的需求，用户在其中能够进行注册、登录、聊天发送图片等功能，聊天功能相对完善，可由用户选择字体等选项。

# 需求分析

## 用户及业务需求

随着互联网的发展，当下即时聊天软件已经发展的十分完善了，然而倘若是在一个没有互联网的局域网中，用户之间想要进行即时通讯则有些不方便，本软件的主要设计目的是提供一个在局域网中能够进行即时通讯的软件。通过一个简单的软件使一个局域网下的用户可以在不使用互联网的情况下进行多人聊天，方便局域网内部的沟通。

图2-1为该软件的用例图。

该图体现了用户通过该软件可进行的操作也一定程度体现了管理员与用户的关系以及管理员在管理软件使用时可以进行的操作等。



图2-1 用例图

## 层次方框图

### 2.2.1用户层次方框图

图2-2位该软件的用户层次方框图。

该图体现了用户从注册到退出中可进行的一系列操作。

用户在开始时可以选择注册或是使用已有的账号直接登录，登陆后直接进入聊天页面，编辑消息时可以对发送出去的消息的字体，颜色以及大小进行设置，随后进行聊天，聊天完成后可退出登录，完成本次聊天。



图2-2 用户层次方框图

### 2.2.2管理员层次方框图

图2-3为该软件的管理员层次方框图。

该图体现了管理员从开始到结束途中可进行的一系列操作。

管理员在开始时首先将服务端开启，之后可选择对用户的数据进行管理或者进入客户端进行聊天。管理用户数据即对数据进行增删改，修改后即可结束，进入客户端聊天与用户的层次方框图类似，不再赘述。

图2-3 管理员层次方框图

## 数据流图

### 2.3.1注册数据流图

图2-4为该软件注册时的数据流图。

图2-5注册数据流图

### 2.3.2登录数据流图

图2-5为该软件登录时的数据流图。

客户端发送登录请求到服务端，登录请求中包括登录时所使用的账户、密码以及登录时所使用的机器的IP地址。登录请求到达服务端后，服务端与数据库中的信息进行比对，给回登录反馈，并将IP地址与数据库中进行比对，如为新的IP地址则将新的IP地址存入数据库。

图2-5登录数据流图

### 2.3.3聊天数据流图

图2-6为该软件聊天时的数据流图。

客户端一发送消息到服务端，发送的消息的内容包括消息文本以及消息文本的字体、大小颜色，服务端接收到消息将其转发给同一局域网内的IP地址在登录时被注册了的IP地址，局域网内在线的其他客户端在收到信息后在客户端中的聊天页面按照对应的文本字体显示，发消息的时间以及发送者的昵称以小号的字体显示在消息的上方。

图2-5聊天数据流图

# 整体设计

## 3.1 客户端设计

### 3.1.1登录页面设计

图3-1为该软件登录页面的设计图

登录页面为软件的起始页面布局，密码输入框设置为不可见，用户可直接选择登陆或先行注册后再进行登录，登录出错会给予相应的提示。

图3-1登录页面布局图

### 3.1.2注册页面设计

图3-2为该软件注册页面的设计图

可从登录页面跳转至注册页面，注册页面可实现用户注册的功能，注册时要求用户确认自己所输入的密码来确保注册时不会误填。在注册的过程中会根据注册时发生的错误进行各种提示。注册完成后用户可返回登录页面进行登录。

图3-2注册页面布局图

### 3.1.3聊天页面设计

图3-3为该软件聊天页面的设计图

聊天框留相对大的面积用来显示聊天信息，下面加入三个下拉菜单栏来设置消息的字体、大小以及颜色，最下方为消息编辑框，用于编辑消息的文本。

图3-3聊天页面布局图

## 3.2 服务端设计

图3-4为该软件服务端页面的设计图

聊天框显示当下的聊天内容，右侧显示所有在线的人员，下方两个按钮，一个开启或暂停服务端的工作，另一个跳转进行人员的信息管理。

图3-3服务端页面布局图

## 3.3程序流程

图3-5为该软件程序流程图

图3-5程序流程图

# 系统实现

## 服务端的实现

### 4.1.1登录功能的实现

核心代码：

验证密码正确与否的方法：

int **confirm**(QString aa,QString bb) //登录验证

{

QSqlDatabase db;

if(QSqlDatabase::contains("qt\_sql\_default\_connection"))

db = QSqlDatabase::database("qt\_sql\_default\_connection");

else

db = QSqlDatabase::addDatabase("QMYSQL");

db.setHostName("127.0.0.1");

db.setPort(3306);

db.setDatabaseName("jiwangkeshe");

db.setUserName("root");

db.setPassword("616709");

db.open();

if(!db.open())

{

qDebug()<<"connect failed"<<"connect to mysql error"<<db.lastError().text();

}

else

{

qDebug()<<"connect to mysql OK";

}

QSqlQuery query(db);

query.exec("select \* from information");

while(query.next()){

//“0”为正确，“1”为密码错误，“2”为用户不存在

if(query.value("username")==aa){

qDebug()<<query.value("username");

if(query.value("password")==bb){

return 0;

}

else if(query.value("password").toString()!=bb){

return 1;

}

}

}

return 2;

}

添加新的登录机器IP到数据库以及返回登录反馈：

QString username = list[1];

QString password = list[2];

QString sign = list[3];

QString address = list[4];

int add = addres(address);

qDebug()<<username<<endl<<password<<endl<<sign;

int judge = confirm(username,password);

qDebug()<<judge;

QString jud = QString::number(judge);

QByteArray sendmsg =jud.toUtf8()+"><"+sign.toUtf8();

QSqlDatabase db;

if(QSqlDatabase::contains("qt\_sql\_default\_connection"))

db = QSqlDatabase::database("qt\_sql\_default\_connection");

else

db = QSqlDatabase::addDatabase("QMYSQL");

db.setHostName("127.0.0.1");

db.setPort(3306);

db.setDatabaseName("jiwangkeshe");

db.setUserName("root");

db.setPassword("616709");

db.open();

if(!db.open())

{

qDebug()<<"connect failed"<<"connect to mysql error"<<db.lastError().text();

}

else

{

qDebug()<<"connect to mysql OK";

}

QSqlQuery query(db);

query.exec("select \* from address");

while(query.next()){

sender->writeDatagram(sendmsg.data(), sendmsg.size(),QHostAddress(query.value("ip").toString()), 45455);

}

### 4.1.2注册功能的实现

核心代码：

QString username = list[1];

QString password = list[2];

QString sign = list[3];

qDebug()<<username<<endl<<password<<endl<<sign;

int judge = registe1(username,password);

qDebug()<<judge;

QString jud = QString::number(judge);

QByteArray sendmsg =jud.toUtf8()+"><"+sign.toUtf8();

QSqlDatabase db;

if(QSqlDatabase::contains("qt\_sql\_default\_connection"))

db = QSqlDatabase::database("qt\_sql\_default\_connection");

else

db = QSqlDatabase::addDatabase("QMYSQL");

db.setHostName("127.0.0.1");

db.setPort(3306);

db.setDatabaseName("jiwangkeshe");

db.setUserName("root");

db.setPassword("616709");

db.open();

if(!db.open())

{

qDebug()<<"connect failed"<<"connect to mysql error"<<db.lastError().text();

}

else

{

qDebug()<<"connect to mysql OK";

}

QSqlQuery query(db);

query.exec("select \* from address");

while(query.next()){

sender->writeDatagram(sendmsg.data(), sendmsg.size(),QHostAddress(query.value("ip").toString()), 45456);

}

}

### 4.1.3聊天功能的实现

核心代码：

receiver->readDatagram(datagram.data(), datagram.size());

qDebug()<<datagram;

QSqlDatabase db;

if(QSqlDatabase::contains("qt\_sql\_default\_connection"))

db = QSqlDatabase::database("qt\_sql\_default\_connection");

else

db = QSqlDatabase::addDatabase("QMYSQL");

db.setHostName("127.0.0.1");

db.setPort(3306);

db.setDatabaseName("jiwangkeshe");

db.setUserName("root");

db.setPassword("616709");

db.open();

if(!db.open())

{

qDebug()<<"connect failed"<<"connect to mysql error"<<db.lastError().text();

}

else

{

qDebug()<<"connect to mysql OK";

}

QSqlQuery query(db);

query.exec("select \* from address");

while(query.next()){

sender->writeDatagram(datagram.data(), datagram.size(),QHostAddress(query.value("ip").toString()), 45457);

}

## 客户端的实现

### 4.2.1登录页面

核心代码：

获取用户名密码以及本机IP并发送：

QHostAddress **getHostIPV4Address**()

{

foreach(const QHostAddress& hostAddress,QNetworkInterface::allAddresses())

if ( hostAddress != QHostAddress::LocalHost && hostAddress.toIPv4Address() )

return hostAddress;

return QHostAddress::LocalHost;

}

QHostAddress address = getHostIPV4Address();

int mes = 1;

QString mesg = QString::number(mes);

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

if(ui->lineEdit->text()!=NULL&&ui->lineEdit\_2->text()!=NULL){

QString signal = QString::number(num);

QByteArray datagram = mesg.toUtf8()+"><"+ui->lineEdit->text().toUtf8()+"><" +ui->lineEdit\_2->text().toUtf8()+"><"+signal.toUtf8()+"><"+address.toString().toUtf8()+"><"+" "+"><"+" ";

sender->writeDatagram(datagram.data(), datagram.size(),QHostAddress("192.168.1.158"), 45454);

}

else{

QMessageBox::information(this,"提示","用户名或密码不能为空");

}

根据登录反馈进行提示以及跳转：

while(receiver->hasPendingDatagrams()) //收到返回消息后的动作

{

QByteArray datagram;

datagram.resize(receiver->pendingDatagramSize());

receiver->readDatagram(datagram.data(), datagram.size());

temp = datagram;

}

QStringList list = temp.split("><");

QString jud = list[0];

QString sign = list[1];

qDebug()<<sign.toInt();

if(num==sign.toInt()){

judge=jud.toInt();

qDebug()<<num;

if(judge==0){

MainWindow \*main =new MainWindow;

main->setWindowTitle("聊天页面");

main->show();

judge=2;

this->close();

}

else if(judge==1){

QMessageBox::information(this,"提示","密码错误");

}

else if(judge==2){

QMessageBox::information(this,"提示","没有该用户");

}

}

### 4.2.2注册页面

核心代码：

获取填写的用户名密码发送：

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

QMessageBox::information(this,"提示","用户名不能为空");

}

else if(ui->lineEdit\_2->text()!=ui->lineEdit\_3->text()){

QMessageBox::information(this,"提示","两次输入密码不一致");

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

ui->lineEdit\_3->setText(NULL);

}

else if(ui->lineEdit\_2->text()==NULL&&ui->lineEdit\_3->text()==NULL){

QMessageBox::information(this,"提示","密码不可以为空");

}

else{

int mes = 2;

QString mesg = QString::number(mes);

QString signal = QString::number(num1);

QByteArray datagram =mesg.toUtf8()+"><"+ ui->lineEdit->text().toUtf8()+"><"+ui->lineEdit\_2->text().toUtf8()+"><" //传输的消息

+signal.toUtf8()+"><"+" "+"><"+" "+"><"+" ";

sender->writeDatagram(datagram.data(), datagram.size(),QHostAddress("192.168.1.158"), 45454);

qDebug()<<datagram<<endl;

qDebug()<<num1<<endl;

}

根据注册反馈进行提示：

while(receiver->hasPendingDatagrams()) //收到消息

{

QByteArray datagram;

datagram.resize(receiver->pendingDatagramSize());

receiver->readDatagram(datagram.data(), datagram.size());

temp1 = datagram;

}

QStringList list = temp1.split("><");

QString jud = list[0];

QString sign = list[1];

qDebug()<<sign.toInt();

if(num1==sign.toInt()){

if(jud.toInt()==0){

QMessageBox::about(NULL, "提示", "注册成功。");

}

else if(jud.toInt()==1){

QMessageBox::about(NULL, "提示", "该用户名已被使用");

}

else

QMessageBox::about(NULL, "提示", "出 错！");

}

### 4.2.3聊天页面

核心代码：

获取当前时间，文本以及字体并发送：

QTime current\_time =QTime::currentTime(); //获取当前时间

QString time = current\_time.toString();

if(ui->textEdit->toPlainText()!=NULL ){

int mes =0;

QString mesg=QString::number(mes);

int ss=ui->comboBox\_2->currentIndex()+1;

QString sss=QString::number(ss);

QByteArray datagram = mesg.toUtf8()+"><" //聊天信息

+user.toUtf8()+"><"

+ui->textEdit->toPlainText().toUtf8()+"><"

+ui->fontComboBox->currentText().toUtf8()+"><"

+sss.toUtf8()+"><"

+ui->comboBox->currentText().toUtf8()+"><"

+time.toUtf8();

sender->writeDatagram(datagram.data(), datagram.size(),QHostAddress("192.168.1.158"), 45454);

ui->textEdit->setText(NULL);

}

else{

QMessageBox::information(this,"提示","发送消息不可为空");

}

接受消息并处理显示：

while(receiver->hasPendingDatagrams())

{

QByteArray datagram;

datagram.resize(receiver->pendingDatagramSize());

receiver->readDatagram(datagram.data(), datagram.size());

temp2 = datagram;

qDebug()<<temp2;

}

QStringList list;

list = temp2.split("><");

QString username = list[1];

QString text = list[2];

QString font = list[3];

int size = list[4].toInt();

QString color = list[5];

QString time =list[6];

ui->textBrowser->append(tr("<font size = 2 color = gray>%1 &nbsp;&nbsp; %2 :</font>").arg(time).arg(username));

//富文本

ui->textBrowser->append(tr("<font size = %4 face = %1 color = %3>%2</font>").arg(font).arg(text).arg(color).arg(size));

# 5 软件测试

## 5.1 登录功能

## 5.2 注册功能

## 5.3 聊天功能

# 总结

## 6.1 已完成的功能模块

本次课程设计基本完成了在开始时老师布置的课程设计的要求，在自己的努力下陆续完成了多人聊天、基于UDP、可视化图形界面这几个比较重要的任务点，其次自己在过程中也尝试使用QT强大的UI编辑功能尝试做了字体，富文本等内容，总的来说还是收获比较大的。

## 6. 2 未完成的功能模块

与预期的结果相去甚远的部分就是服务端的界面设计了，最终服务端只有一个简陋的页面，没有做出一个能够让管理员真真正正可以去进行管理的服务端，服务端到最终也只是充当一个校验登录与注册结果、收发信息的中转站，没有能够实现一个服务端应有的功能。

其次就是在客户端的设计上，整体的界面还是过于简陋，虽然QT有强大的UI编辑的能力，但是由于自己初学，仍有许多地方不得要领，还是由许多值得学习的东西都等着我去突破。

## 6. 3个人小结

由于本次自己所选的题目是基于UDP的聊天软件，所以在准备课程设计的时间里花费了不少精力去了解和学习传输层协议的资料，更加深入的了解了TCP/IP协议、UDP协议等的实现机制，明白了他们的具体的工作流程，加深了自己对于这门学科的理解。同时由于自己是第一次尝试使用QT来进行程序的编写，在完成课程设计的过程中总是不断地遇到问题，只有不断地学习才能将进度推展下去，通过一次次的解决问题，我对于QT的掌握速度还相对较快，为自己又积累了一项技能。

## 6.2 展望

# 参考文献

[1] 陆文周. Qt5开发及实例[M]，北京：电子工业出版社，2014：473~568

[2] 霍亚飞. Qt Creator快速入门 第2版[M].，北京：冶金工业出版社，2007：207~362

[3] 霍亚飞. qt creator快速入门 第3版[M]. 北京：清华大学出版社，2003：152~203

[5] 刘腾红，孙细明.信息系统分析与设计[M]. 北京：科学出版社，2003：25~116

[6] 塔嫩鲍姆,|计算机网络（英文版第5版）.机械工业出版社,2011：13~52