**设计思路：**

传输控制协议（TCP，Transmission Control Protocol）是一种面向连接的、可靠的、基于字节流的传输层通信协议，由IETF的RFC 793[1]  定义。

TCP旨在适应支持多网络应用的分层协议层次结构。 连接到不同但互连的计算机通信网络的主计算机中的成对进程之间依靠TCP提供可靠的通信服务。TCP假设它可以从较低级别的协议获得简单的，可能不可靠的数据报服务。 原则上，TCP应该能够在从硬线连接到分组交换或电路交换网络的各种通信系统之上操作。

**源代码：**

#ifndef MYSERVER\_H

#define MYSERVER\_H

#include <QTcpServer>

#include "mytcpsocket.h"

class **MyServer** : public QTcpServer

{

Q\_OBJECT

public:

**MyServer**(QObject \*parent = 0);

QList<MyTcpSocket \*> tcpSocketList;

signals:

void **updateServer**(QString);

public slots:

void **updateClients**(QString,int);

void **slotDisconnected**(int);

protected:

void ***incomingConnection***(qintptr socketDescriptor);

};

#endif // MYSERVER\_H

#include "myserver.h"

MyServer::**MyServer**(QObject \*parent)

:QTcpServer(*parent*)

{

listen(QHostAddress::Any,8888);

}

//出现一个新的连接时触发

void MyServer::***incomingConnection***(qintptr socketDescriptor)

{

MyTcpSocket \*tcpClientSocket = new MyTcpSocket(this);

connect(tcpClientSocket, SIGNAL(updateClients(QString,int)), this, SLOT(updateClients(QString,int)));

connect(tcpClientSocket, SIGNAL(disconnected(int)), this, SLOT(slotDisconnected(int)));

tcpClientSocket->*setSocketDescriptor*(socketDescriptor);

tcpSocketList.append(tcpClientSocket);

qDebug() << "成功连接";

}

//将任意客户端发来的信息进行广播

void MyServer::**updateClients**(QString msg,int descriptor)

{

emit updateServer(msg);

for(int i = 0; i < tcpSocketList.count() ; i ++)

{

QTcpSocket \*socket = tcpSocketList.at(i);

if(socket->*socketDescriptor*() != descriptor)

{

socket->write(msg.toStdString().c\_str());

}

// socket->write(msg.toStdString().c\_str());

// qDebug() << "已经写出";

}

// qDebug() << msg;

}

//将断开连接的TcpSocket对象删除

void MyServer::**slotDisconnected**(int descriptor)

{

for(int i = 0 ; i < tcpSocketList.count();i++)

{

QTcpSocket \*socket = tcpSocketList.at(i);

if(socket->*socketDescriptor*() == descriptor)

{

tcpSocketList.removeAt(i);

return;

}

}

return;

}

#ifndef MYTCPSOCKET\_H

#define MYTCPSOCKET\_H

#include <QTcpSocket>

class **MyTcpSocket** : public QTcpSocket

{

Q\_OBJECT

public:

**MyTcpSocket**(QObject \*parent=0);

signals:

void **updateMainWindow**(QString);

void **updateClients**(QString,int);

void **disconnected**(int);

protected slots:

void **dataReceived**();

void **slotDisconnected**();

};

#endif // MYTCPSOCKET\_H

#include "mytcpsocket.h"

MyTcpSocket::**MyTcpSocket**(QObject \*parent)

{

connect(this,SIGNAL(readyRead()),this,SLOT(dataReceived()));

connect(this,SIGNAL(disconnected()),this,SLOT(slotDisconnected()));

}

void MyTcpSocket::**dataReceived**()

{

QByteArray array = readAll();

QString msg = array;

// qDebug() << msg;

emit updateMainWindow(msg);

emit updateClients(msg,*socketDescriptor*());

}

void MyTcpSocket::**slotDisconnected**()

{

emit disconnected(*socketDescriptor*());

}