头文件

```
#include <QFileDialog>
#include <QMessageBox>
#include <QTextStream>
#include <QDataStream>
#include <QDebug>
#include <QDir>
#include <QObject>
#include <QTextCodec>
#include <QPainter>
#include <QThread>
#include <QHostAddress>
#include <QMutex>

//#include <QTcpSocket>
//#include <QUdpSocket>
//#include<QMediaPlayer>
```

Tcp通信

服务器

```
//构造
connect(tcpServer,SIGNAL(newConnection()),this,SLOT(onNewConnection()));
//开始监听
quint16 port=8000;//端口
tcpServer->listen(QHostAddress::LocalHost,port);
//onNewConnection
tcpSocket = tcpServer->nextPendingConnection(); //创建socket
//停止监听
if (tcpServer->isListening())
        tcpServer->close();
```

客户端

```
//开始连接
quint16 port=8000;//端口
tcpServer->listen(QHostAddress::LocalHost,port);
//断开连接
 if (tcpClient->state()==QAbstractSocket::ConnectedState)
        tcpClient->disconnectFromHost();
```

读写

```
//发送
QString msg;
QByteArray str=msg.toUtf8();
//str.append("\n);
tcpClient->write(str);
//接收
  while(tcpClient->canReadLine()){
       QString str = tcpClient->readLine();
       //do something
}
```

Udp通信

发送端

```
//构造
socket = new QUdpSocket;
socket->bind(23333);

quint16 port=8000;
QString msg;
QByteArray str=msg.toUtf8();
socket->writeDatagram(str,QHostAddress::Broadcast,port);//广播
```

接收端

```
  socket=new QUdpSocket;
  socket->bind(8000);
  connect(socket,&QUdpSocket::readyRead,[this]{
        while(socket->hasPendingDatagrams()){
            QByteArray datagram;
            datagram.resize(socket->pendingDatagramSize());
            socket->readDatagram(datagram.data(),datagram.size());
            QString str=datagram.data();
            //QStringList list=str.split(" ");
            //do something
        }
  });
```

对话框

信息框

```
QMessageBox::information(this, "消息框", "");
QMessageBox::warning(this, "警告", "")
```

打开/保存文件

```
QString FileName = QFileDialog::getOpenFileName(this,"Open",QDir::currentPath(),"all(*.*)");
if(!FileName.isEmpty()){
    QFile aFile(FileName);
    if (!aFile.exists()||!aFile.open(QIODevice::ReadOnly))
        return;
    QTextStream s(&aFile);
    //do something
    aFile.close();
}
```

```
QString FileName=QFileDialog::getSaveFileName(this,"Save",QDir::currentPath(),"all(*.*)");
if (!FileName.isEmpty()){
    QFile aFile(FileName);
    if (!aFile.open(QIODevice::WriteOnly))
        return false;
    QTextStream s(&aFile);
    //do something
    aFile.close();
    return;
}
```

也可以使用C++风格的文件读写。

IP地址获取

```
QString MainWindow::getLocalIP()
{//获取本机IPv4地址
    QString hostName=QHostInfo::localHostName();//本地主机名
    QHostInfo   hostInfo=QHostInfo::fromName(hostName);
    QString   localIP="";

    QList<QHostAddress> addList=hostInfo.addresses();//

    if (!addList.isEmpty())
    for (int i=0;i<addList.count();i++)
    {
        QHostAddress aHost=addList.at(i);
        if (QAbstractSocket::IPv4Protocol==aHost.protocol())
        {
            localIP=aHost.toString();
            break;
        }
    }
    return localIP;
}
```

分段发送文件

```
//发送
  QFile f(fileName);
    f.open(QFile::ReadOnly);

    QByteArray arr;
    QDataStream out(&arr,QIODevice::WriteOnly);
    qint64 size=f.size();
    out<<size;
```

```
    socket->write(arr);
    socket->waitForBytesWritten();
    for (int cur=0;cur<size;) {
        QByteArray block=f.read(40960);
        cur += block.size();
        socket->write(block);
        if (!socket->waitForBytesWritten(100))
                    break;
    }
    f.close();
//接受
qint64 size,cur=0;
while (cur < size){
           socket->waitForReadyRead(150);
           QByteArray arr = socket->readAll();
           cur += arr.size();
       }
```

继承QObject，使用moveToThread()，用信号与槽控制线程。

继承QTcpServer重写incommingConnect，实现多线程

注意：不同线程之间connect，不要用lambda表达式