C#编程中数据流的使用一直不很熟练，没有一个系统的认识，但是它的重要性显然不言而喻。System.IO下的Stream类是所有数据流的基类，当我们对数据进行逐字节操作时，首先需要将数据转换为数据流。C#数据流主要分为三类：FileStream、MemoryStream、NetworkStream，还有常用的StreamReader、StreamWriter和TextWriter类等。

**1.FileStream（文件流），继承Stream类。**

　　由名字可以看到这个类主要是对文件进行操作的。得到FileStream对象的常用方式有如下两种：

　　　　FileStream fs=new FileStream(string path,FileMode mode,FileAccess access);//利用构造函数创建，FileMode枚举和FileAccess枚举这里就不总结了。

　　　　FileStream fs=File.OpenWrite(@”C:\”);//还有File.Open、File.OpenRead、File.OpenText都用的很多。

　　 得到FileStream对象后，调用常见的Read和Write方法即可进行读写，如果要读写比1kb字节大的文件时可以加个while循环来获得数据。

//写文件流

string str = "hh";

byte[] bytes1 = new byte[1024];

bytes1 = Encoding.UTF8.GetBytes(str);

FileStream fst = new FileStream("sd", FileMode.OpenOrCreate, FileAccess.Write);

fst.Write(bytes1, 0, bytes1.Length);

//读文件流

FileStream fs = File.OpenRead("sd");

byte[] bytes2 = new byte[1024];

int num = fs.Read(bytes2, 0, bytes2.Length);

**2.MemoryStream，内存流**

　　它是以内存流的形式对内存中的字节数组进行操作，支持对数据流的查找和随机访问，和文件流一样可以使用Position属性得到内存流的当前位置。内存流的优点是对长度不定的数据进行缓存时，由于内存流是可以自动增长的，因此使用很方便。

byte[] bytes=Encoding.UTF8.GetBytes("hh");

using(MemoryStream ms=new MemoryStream())

{

　　　ms.Write(bytes, 0, bytes.Length);

　　　byte[] byte2 = new byte[bytes.Length];

　　　ms.Position = 0;

　　　int n = ms.Read(byte2, 0, byte2.Length);

}

**3.NetworkStream，网络流**

　　这个流在网络编程里使用非常普遍。当客户端程序与服务器程序进行通信时数据的接受与发送就要用到这个类，有一点要特别注意，NetworkStream类仅仅只支持面向连接的套接字，比如使用TcpClient编写聊天程序时，只有服务器与客户端的TcpClient进行了连接后才能使用NetworkStream。得到NetworkStream对象创建好后就可以利用网络流对象进行网络数据的传递，但是网络流对象是没有Position对象的，无法指定起始位置来进行读写。还有NetworkStream对象的CanSeek属性始终返回false，使用Position和Seek方法会出现异常。

　　得到NetworkStream对象有2种方式，一种是利用socket对象获得，一种是利用TcpClient对象来获得，后者我用的非常多。

//写入数据到缓冲流

NetworkStream ns = new NetworkStream(socket);

NetworkStream ns = tcpClient.GetStream();

if (ns.CanWrite)

{

byte[] bytes = Encoding.UTF8.GetBytes("hh");

ns.Write(bytes, 0, bytes.Length);

}

//读取数据

if (ns.CanRead)

{

int num;

byte[] bytes = new byte[1024];

while (ns.DataAvailable)

{

num = ns.Read(bytes, 0, bytes.Length);

}

}

　　很明显，在实际编程中如果使用字节数组来接受或发送数据太麻烦了，后来从网络编程老师那学到了两个非常好用的类BinaryReader类和BinaryWriter类。我们在实际通信中使用最多的还是字符串通信，只要得到NetworkStream对象后，就可以得到字符串读取流和写入流，比如以下代码

TcpClient tcp = new TcpClient();

NetworkStream networkStream = new NetworkStream();

BinaryReader reader;

BinaryWriter writer;

string msg = "hh";

networkStream = tcp.GetStream();

writer = new BinaryWriter(networkStream);

writer.Write(msg);//写字符串

string str = reader.ReadString();//读字符串

还有StreamReader类和StreamWriter类用的也是比较多，比如读一个txt文档读到末尾，可以调用StreamReader对象的ReadToEnd方法来拿到这个txt的所有字符串内容。另外如果要对发送的数据进行加密的话，有一个CryptoStream类用起来也挺方便的，在创建这个对象时可以直接就指定采用的对称加密算法，比如DES加密算法。解密时也非常方便，只需要使用和加密时相同的密钥创建CryptoStream实例，并在创建实例时指定CryptoStreamMode为读模式，即可拿到解密后的数据。

https://blog.csdn.net/luanpeng825485697/article/details/78165813