C# FileStream和StreamWriter用法

 FielStream类操作的是字节和字节数组，而stream类操作的是字符数据。字符数据易于使用，但是有些操作，比如随机文件访问(访问文件中间某点的数据)，就必须由FileStream对象执行。  
   FielStream类维护内部文件指针，该指针指向文件中进行下一次读写操作的位置。  
   FileStream只能处理原始字节(raw byte),处理原始字节的功能使用FileStream类可以用于处理任何数据文件。  
   通过读取字节数据，FileStream对像可以用于读取图像和声音。代价是FileStream类不能将数据直接读入字符串  
   此指针可以修改用Seek方法，它有两个参数：  
   第一个参数：规定文件指针以字节为参数的移动距离  
   第二个参数：规定开始计算的起始位置

        byte[] byDataValue = new byte[200];  
        char[] charDataValue = new char[200];

        /// <summary>  
        /// FilesStream读取用法  
        /// </summary>  
        private void FilesStreamReadFile()  
        {  
            try  
            {  
                FileStream fsFile = new FileStream(@"d:\log.cs", FileMode.Open);  
                //文件指针移到文件的135个字节  
                fsFile.Seek(135, SeekOrigin.Begin);  
                //将接下来的字节读到Array中  
                fsFile.Read(byDataValue, 0, 200);  
            }  
            catch (Exception e)  
            {  
                throw e;  
            }  
            //将字节转换成字符  
            Decoder dc = Encoding.UTF8.GetDecoder();  
            //字节数组转换成字符数组，便于显示  
            dc.GetChars(byDataValue, 0, byDataValue.Length, charDataValue, 0);

        }

        /// <summary>  
        /// FilesStream写入用法  
        /// </summary>  
        private void FileStreamWriteFile()  
        {  
            FileStream fsFile = new FileStream(@"D:\log.txt", FileMode.Create);  
            charDataValue="This is test string".ToCharArray();  
            byDataValue=new byte[charDataValue.Length];

            //将字符数组转换成字节数组  
            Encoder ec=Encoding.UTF8.GetEncoder();  
            ec.GetBytes(charDataValue,0,charDataValue.Length,byDataValue,0,true);

            //将指针设定起始位置  
            fsFile.Seek(0,SeekOrigin.Begin);  
            //写入文件  
            fsFile.Write(byDataValue,0,byDataValue.Length);  
        }  
    FileStream对象通常会在StreamWriter或StreamReader内使用，并使用它们的方法处理文件。  
        /// <summary>  
        /// StreamWriter写入方法  
        /// </summary>  
        private void StreamWriterMetod()  
        {  
            try  
            {  
                FileStream fsFile = new FileStream(@"d:\log.txt",FileMode.OpenOrCreate);  
                StreamWriter swWriter = new StreamWriter(fsFile);  
                //写入数据  
                swWriter.WriteLine("Hello Wrold.");  
                swWriter.WriteLine("It is now {0}", DateTime.Now.ToLongDateString());     
                swWriter.Close();  
            }  
            catch (Exception e)  
            {

                throw e;  
            }  
        }

        /// <summary>  
        /// StreamWriter读取方法  
        /// </summary>  
        private void StreamReaderMetod()  
        {  
            string sLine = "";  
            try  
            {  
                FileStream fsFile = new FileStream(@"d:\log.txt", FileMode.Open);  
                StreamReader srReader = new StreamReader(fsFile);  
                //读取文件(读取大文件时，最好不要用此方法)  
                sLine = srReader.ReadToEnd();  
                txbValue.Text = sLine;  
                srReader.Close();  
            }  
            catch (Exception e)  
            {  
                throw e;  
            }  
        }  
        /// <summary>  
        /// StreamWriter读取方法2  
        /// </summary>  
        private void StreamReaderMetodTwo()  
        {  
            try  
            {  
                FileStream fsFile = new FileStream(@"d:\log.txt", FileMode.Open);

                StreamReader srReader = new StreamReader(fsFile);  
                int iChar;  
                iChar = srReader.Read();  
                while (iChar != -1)  
                {  
                    txbValue.Text += (Convert.ToChar(iChar));  
                    iChar = srReader.Read();  
                }  
                srReader.Close();  
            }  
            catch (Exception e)  
            {  
                throw e;  
            }  
        }  
注：FileMode枚举成员值在文件存在与不存在时的情况

Append  
 文件存在时：打开文件，流指向文件的末尾，只能与枚举FileAcess.Write联合使用  
 文件不存在时：创建一个新文件。只能与枚举FileAcess.Write联合使用  
   
Create  
 文件存在时：删除该文件，然后创建新文件  
 文件不存在时：创建新文件  
   
CreateNew  
 文件存在时：抛出异常  
 文件不存在时：创建新文件  
   
Open  
 文件存在时：打开现有文件，流指向文件开头  
 文件不存在时：抛出异常  
   
OpenOrCreate  
 文件存在时：打开文件，流指向文件开头  
 文件不存在时：创建新文件  
   
Truhncate  
 文件存在时：打开现有文件，清除其内容。流指向文件开头，保留文件的初始创建日期。  
 文件不存在时：抛出异常