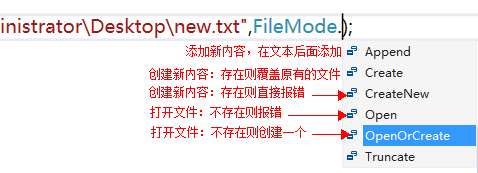
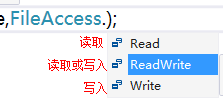
**FileStream 文件流 非静态**

1. FileStream与File的区别：

好比两个水桶，一个装满水，一个没装水，File的使用就是直接把水桶的水全部倒进没水的水桶里，FileStream的使用则是用个水瓢一勺一勺的把水装进水桶，这样就造成了File类没法对大数据进行操作，因为内存无法承受负担

1. 由于是非静态类，所以使用的时候我们需要对其实例化：

微信截图_20171214162121

1. **FileStream** **fileReadWrite** = **new FileStream**（**string path**，**FileMode mode**，**FileAccess access**）;
2. **紫色：实例名；**
3. **黑色：实例化；**
4. **绿色：需要创建或者读取的文件的路径（绝对路径和相对路径）**
5. **红色：对文件的操作，打开、追加、创建等，根据自己选择**
6. **蓝色：对文件中的内容操作，读取、写入等**

**四） 当需要读取数据的时候：蓝色为：Read**

**第一步：**

先进行实例化；

微信截图_20171214164536

**第二步：**

定义一个字节数组（byte[]）通过 第三步 在 流 读取的字节用来保存在这个数组里面；

微信截图_20171214164755

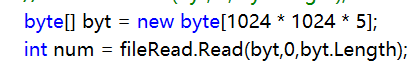
**（图中读取的字节为 5MB）**

**第三步：**

**int** fileRead.Read（**byte[] array**，**int offset**，**int count**）；

1. **通过以上方法 把 流 中读取的字节放入byte[]数组里面**
2. **并且从字节int offset开始读取：由0开始**
3. **至int count结束：写方法2中定义的字节数组的长度**
4. **返回一个int来表示本次读取，实际读取到的字节数（不包括null）**

**代码：**

****

**第四步：**

string Encoding.编码类型.GetString(byte[])；

通过该方法把字节数组转换成字符串，然后进行输出，就能看到读取的内容；

**注意：**

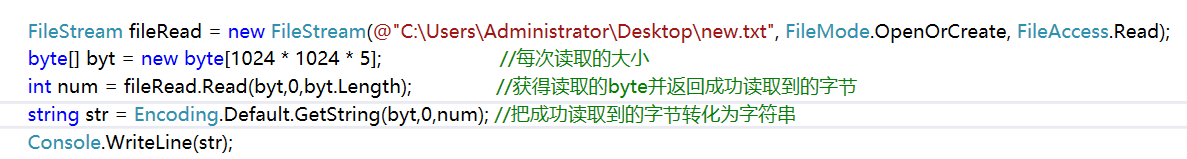
1. **读取的编码类型需要跟文件的编码类型相同，否则乱码**
2. **由于第三步中，我们读取的范围由0字节开始到字节数组的长度结束，也就是读取5MB，但是，如果我们的一个文件的大小小于这个读取的大小，这时候我们直接转换成string类型，就会出现很多空字符，因为我们读取了5MB，但是文件只有7KB，就好比一个很大的勺子，装了一滴水，那么就剩下很多部分，这时候我们就要用以下方法来解决：**



**（0表示从0个字节开始，num就表示 第三步 中，本次读取成功读取到的字节数）**

**第五步：**

完整代码图：



**五）当需要写入数据的时候：蓝色为：Write**

**第一步：**

先进行实例化；

微信截图_20171214164536

**第二步：**

byte[] Encoding.编码类型.GetBytes(string value);

通过该方法，把想写入文件的内容转换成byte[]字节数组

微信截图_20171214191526

**第三步：**

fileWrite.Write（**byte[] array**，**int offset**，**int count**）；

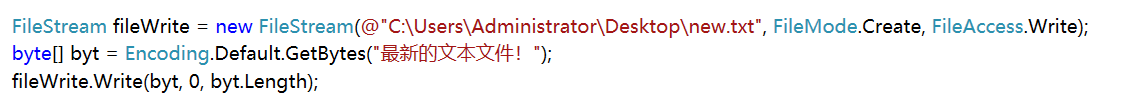
**ar蓝色：为第二步转换的byte数组**

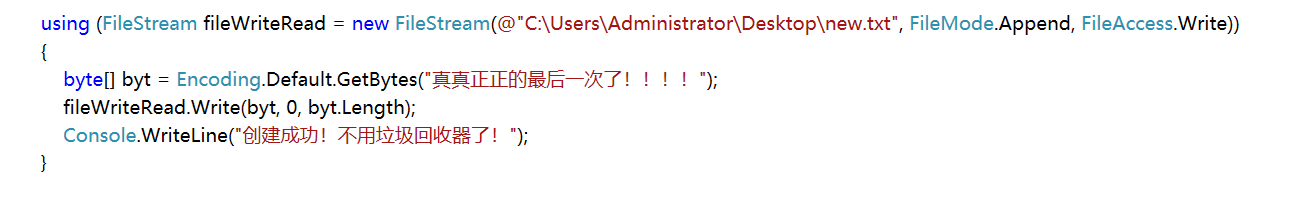
**红色：写入的是后从第0个字节开始**

**绿色：写入字节的长度**

**第四步：**

**完整代码：**



1. **每次程序结束的时候我们需要自己写个资源释放！**
2. 程序结尾写上
3.  使用using()

{

}