

# 1 System.IO.StreamReader Class

```
2 [ILAsm]  
3 .class public serializable StreamReader extends System.IO.TextReader  
4 [C#]  
5 public class StreamReader: TextReader
```

## 6 Assembly Info:

- 7 • *Name:* mscorlib
- 8 • *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- 9 • *Version:* 2.0.x.x
- 10 • *Attributes:*
  - 11 ○ CLSCompliantAttribute(true)

## 12 Implements:

- 13 • **System.IDisposable**

## 14 Summary

15 Implements a `System.IO.Stream` that reads characters from a byte stream in a  
16 particular encoding.

## 17 Inherits From: System.IO.TextReader

18  
19 **Library:** BCL

20  
21 **Thread Safety:** All public static members of this type are safe for multithreaded operations.  
22 No instance members are guaranteed to be thread safe.

## 24 Description

25 The `System.IO.StreamReader` class is designed for character input in a particular  
26 `System.Text.Encoding`, whereas subclasses of `System.IO.Stream` are designed for  
27 byte input and output.

28  
29 [*Note:* `System.IO.StreamReader` defaults to UTF-8 encoding unless specified otherwise,  
30 instead of defaulting to the ANSI code page for the current system. UTF-8 handles  
31 Unicode characters correctly and provides consistent results on localized versions of the  
32 operating system.

33  
34 When reading from a `System.IO.Stream`, it is more efficient to use a buffer that is the  
35 same size as the internal buffer of the stream.

36  
37 By default, a `System.IO.StreamReader` is not thread safe. For a thread-safe wrapper,  
38 see `System.IO.TextReader.Synchronized`.

1  
2 ]  
3

# 1 StreamReader(System.String, 2 System.Text.Encoding, System.Boolean, 3 System.Int32) Constructor

```
4 [ILAsm]  
5 public rtspecialname specialname instance void .ctor(string path, class  
6 System.Text.Encoding encoding, bool detectEncodingFromByteOrderMarks,  
7 int32 bufferSize)
```

```
8 [C#]  
9 public StreamReader(string path, Encoding encoding, bool  
10 detectEncodingFromByteOrderMarks, int bufferSize)
```

## 11 Summary

12 Constructs and initializes a new instance of the `System.IO.StreamReader` class for the  
13 specified file name, with the specified character encoding, byte order mark detection  
14 option, and buffer size.

## 15 Parameters

Parameter	Description
<i>path</i>	A <code>System.String</code> that specifies the complete file path to read.
<i>encoding</i>	A <code>System.Text.Encoding</code> that specifies the character encoding to use.
<i>detectEncodingFromByteOrderMarks</i>	A <code>System.Boolean</code> value that indicates whether the new <code>System.IO.StreamReader</code> is required to look for byte order marks at the beginning of the stream. Specify <code>true</code> to enable detection of byte order marks; otherwise, specify <code>false</code> .
<i>bufferSize</i>	A <code>System.Int32</code> that specifies the minimum buffer size, in number of 16-bit characters. If less than the minimum allowable size (128 characters), the minimum allowable size is used.

## 16 17 Description

18 This constructor initializes the `System.IO.StreamReader.CurrentEncoding` property  
19 using *encoding*.

20

1 If requested, the current constructor detects the encoding by examining the first three  
2 bytes of the stream. The constructor automatically recognizes UTF-8, little-endian  
3 Unicode, and big-endian Unicode text if the file starts with the appropriate byte order  
4 marks. Otherwise, the user-provided encoding is used. See the  
5 `System.Text.Encoding.GetPreamble` method for more information.

6  
7 [Note: *path* is not required to be a file stored on disk; it can be any part of a system  
8 that supports access via streams. For example, depending on the system, this class  
9 might be able to access a physical device.

10  
11 When reading from a `System.IO.Stream`, it is more efficient to use a buffer that is the  
12 same size as the internal buffer of the stream.

13  
14 For information on the valid format and characters for path strings, see  
15 `System.IO.Path`.

16 ]  
17

18 **Exceptions**

Exception	Condition
<b>System.IO.IOException</b>	<i>path</i> is in an invalid format or contains invalid characters.
<b>System.IO.DirectoryNotFoundException</b>	The directory information specified in <i>path</i> was not found.
<b>System.IO.FileNotFoundException</b>	The file specified in <i>path</i> was not found.
<b>System.ArgumentException</b>	<i>path</i> is an empty string ("").
<b>System.ArgumentNullException</b>	<i>path</i> or <i>encoding</i> is null.
<b>System.ArgumentOutOfRangeException</b>	<i>buffersize</i> is less than or equal to zero.

19

20

# 1 StreamReader(System.String, 2 System.Text.Encoding, System.Boolean) 3 Constructor

```
4 [ILAsm]  
5 public rtspecialname specialname instance void .ctor(string path, class  
6 System.Text.Encoding encoding, bool detectEncodingFromByteOrderMarks)  
  
7 [C#]  
8 public StreamReader(string path, Encoding encoding, bool  
9 detectEncodingFromByteOrderMarks)
```

## 10 Summary

11 Constructs and initializes a new instance of the `System.IO.StreamReader` class for the  
12 specified file name, with the specified character encoding and byte order mark detection  
13 option.

## 14 Parameters

Parameter	Description
<i>path</i>	A <code>System.String</code> that specifies the complete file path to read.
<i>encoding</i>	A <code>System.Text.Encoding</code> that specifies the character encoding to use.
<i>detectEncodingFromByteOrderMarks</i>	A <code>System.Boolean</code> value that indicates whether the new <code>System.IO.StreamReader</code> is required to look for byte order marks at the beginning of the stream. Specify <code>true</code> to enable detection of byte order marks; otherwise, specify <code>false</code> .

15

## 16 Description

17 This constructor initializes the `System.IO.StreamReader.CurrentEncoding` property  
18 using *encoding*, and the internal buffer to the default size. [*Note:* The default buffer size  
19 is implementation defined.]  
20

21

22

23 If requested, the current constructor detects the encoding by examining the first three  
24 bytes of the stream. The constructor automatically recognizes UTF-8, little-endian  
25 Unicode, and big-endian Unicode text if the file starts with the appropriate byte order  
26 marks. Otherwise, the user-provided encoding is used. See the

1     System.Text.Encoding.GetPreamble method for more information.  
2  
3     [Note: *path* is not required to be a file stored on disk; it can be any part of a system  
4     that supports access via streams. For example, depending on the system, this class  
5     might be able to access a physical device.  
6  
7     For information on the valid format and characters for path strings, see  
8     System.IO.Path.  
9  
10    ]

11 **Exceptions**

Exception	Condition
<b>System.IO.IOException</b>	<i>path</i> is in an invalid format or contains invalid characters.
<b>System.IO.DirectoryNotFoundException</b>	The directory information specified in <i>path</i> was not found.
<b>System.IO.FileNotFoundException</b>	The file specified in <i>path</i> was not found.
<b>System.ArgumentException</b>	<i>path</i> is an empty string ("").
<b>System.ArgumentNullException</b>	<i>path</i> or <i>encoding</i> is null.

12

13

# 1 StreamReader(System.String, 2 System.Text.Encoding) Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(string path, class  
5 System.Text.Encoding encoding)  
  
6 [C#]  
7 public StreamReader(string path, Encoding encoding)
```

## 8 Summary

9 Constructs and initializes a new instance of the `System.IO.StreamReader` class for the  
10 specified file name and with the specified character encoding.

## 11 Parameters

Parameter	Description
<i>path</i>	A <code>System.String</code> that specifies the complete file path to read.
<i>encoding</i>	A <code>System.Text.Encoding</code> that specifies the character encoding to use.

## 12 13 Description

14 This constructor initializes the `System.IO.StreamReader.CurrentEncoding` property  
15 using *encoding*, and the internal buffer to the default size. [*Note*: The default buffer size  
16 is implementation defined.]

17  
18  
19  
20 [*Note*: *path* is not required to be a file stored on disk; it can be any part of a system  
21 that supports access via streams. For example, depending on the system, this class  
22 might be able to access a physical device.

23  
24 For information on the valid format and characters for path strings, see  
25 `System.IO.Path`.

26  
27 ]

## 28 Exceptions

Exception	Condition
<b>System.IO.IOException</b>	<i>path</i> is in an invalid format or contains invalid characters.

<b>System.IO.DirectoryNotFoundException</b>	The directory information specified in <i>path</i> was not found.
<b>System.IO.FileNotFoundException</b>	The file specified in <i>path</i> was not found.
<b>System.ArgumentException</b>	<i>path</i> is an empty string ("").
<b>System.ArgumentNullException</b>	<i>path</i> or <i>encoding</i> is null.

1

2

# StreamReader(System.String, System.Boolean) Constructor

```
[ILAsm]
public rtspecialname specialname instance void .ctor(string path, bool
detectEncodingFromByteOrderMarks)

[C#]
public StreamReader(string path, bool detectEncodingFromByteOrderMarks)
```

## Summary

Constructs and initializes a new instance of the `System.IO.StreamReader` class for the specified file name, with the specified byte order mark detection option.

## Parameters

Parameter	Description
<i>path</i>	A <code>System.String</code> that specifies the complete file path to read.
<i>detectEncodingFromByteOrderMarks</i>	A <code>System.Boolean</code> value that indicates whether the new <code>System.IO.StreamReader</code> is required to look for byte order marks at the beginning of the stream. Specify <code>true</code> to enable detection of byte order marks; otherwise, specify <code>false</code> .

## Description

This constructor initializes the `System.IO.StreamReader.CurrentEncoding` property to `System.Text.UTF8Encoding`, and the internal buffer to the default size. [Note: The default buffer size is implementation defined.]

If requested, the current constructor detects the encoding by examining the first three bytes of the stream. The constructor automatically recognizes UTF-8, little-endian Unicode, and big-endian Unicode text if the file starts with the appropriate byte order marks. Otherwise, UTF-8 encoding is used. See the `System.Text.Encoding.GetPreamble` method for more information.

[Note: *path* is not required to be a file stored on disk; it can be any part of a system that supports access via streams. For example, depending on the system, this class might be able to access a physical device.

For information on the valid format and characters for path strings, see

1 System.IO.Path.  
2  
3 ]

#### 4 Exceptions

Exception	Condition
<b>System.IO.IOException</b>	<i>path</i> is in an invalid format or contains invalid characters.
<b>System.IO.DirectoryNotFoundException</b>	The directory information specified in <i>path</i> was not found.
<b>System.IO.FileNotFoundException</b>	The file specified in <i>path</i> was not found.
<b>System.ArgumentException</b>	<i>path</i> is an empty string ("").
<b>System.ArgumentNullException</b>	<i>path</i> is null.

5

6

# 1 StreamReader(System.String) Constructor

```
2 [ILAsm]  
3 public rtspecialname specialname instance void .ctor(string path)  
4 [C#]  
5 public StreamReader(string path)
```

## 6 Summary

7 Constructs and initializes a new instance of the `System.IO.StreamReader` class for the  
8 specified file name.

## 9 Parameters

Parameter	Description
<i>path</i>	A <code>System.String</code> that specifies the complete file path to read.

10

## 11 Description

12 This constructor initializes the `System.IO.StreamReader.CurrentEncoding` property to  
13 `System.Text.UTF8Encoding`, and the internal buffer to the default size. [*Note:* The  
14 default buffer size is implementation defined.]

15

16

17

18 [*Note:* *path* is not required to be a file stored on disk; it can be any part of a system  
19 that supports access via streams. For example, depending on the system, this class  
20 might be able to access a physical device.

21

22 For information on the valid format and characters for path strings, see  
23 `System.IO.Path`.

24

25 ]

## 26 Exceptions

Exception	Condition
<b>System.IO.IOException</b>	<i>path</i> is in an invalid format or contains invalid characters.
<b>System.IO.DirectoryNotFoundException</b>	The directory information specified in <i>path</i> was not found.

<b>System.IO.FileNotFoundException</b>	The file specified in <i>path</i> was not found.
<b>System.ArgumentException</b>	<i>path</i> is an empty string ("").
<b>System.ArgumentNullException</b>	<i>path</i> is null.

1

2

# 1 StreamReader(System.IO.Stream, 2 System.Text.Encoding, System.Boolean, 3 System.Int32) Constructor

```
4 [ILAsm]  
5 public rtspecialname specialname instance void .ctor(class  
6 System.IO.Stream stream, class System.Text.Encoding encoding, bool  
7 detectEncodingFromByteOrderMarks, int32 bufferSize)  
  
8 [C#]  
9 public StreamReader(Stream stream, Encoding encoding, bool  
10 detectEncodingFromByteOrderMarks, int bufferSize)
```

## 11 Summary

12 Constructs and initializes a new instance of the `System.IO.StreamReader` class for the  
13 specified stream, with the specified character encoding, byte order mark detection  
14 option, and buffer size.

## 15 Parameters

Parameter	Description
<i>stream</i>	The <code>System.IO.Stream</code> to read.
<i>encoding</i>	A <code>System.Text.Encoding</code> that specifies the character encoding to use.
<i>detectEncodingFromByteOrderMarks</i>	A <code>System.Boolean</code> value that indicates whether the new <code>System.IO.StreamReader</code> is required to look for byte order marks at the beginning of the stream. Specify <code>true</code> to enable detection of byte order marks; otherwise, specify <code>false</code> .
<i>bufferSize</i>	A <code>System.Int32</code> that specifies the minimum buffer size, in number of 16-bit characters. If <i>bufferSize</i> is less than the minimum allowable size (128 characters), the minimum allowable size is used.

## 16 17 Description

18 This constructor initializes the `System.IO.StreamReader.CurrentEncoding` property  
19 using *encoding* parameter the `System.IO.StreamReader.BaseStream` property using  
20 *stream*.  
21

1 If requested, this constructor detects the encoding by examining the first three bytes of  
2 the stream. The constructor automatically recognizes UTF-8, little-endian Unicode, and  
3 big-endian Unicode text if the file starts with the appropriate byte order marks.  
4 Otherwise, the user-provided encoding is used. For more information, see the  
5 `System.Text.Encoding.GetPreamble` method.

6  
7 [Note: When reading from a `System.IO.Stream`, it is more efficient to use a buffer that  
8 is the same size as the internal buffer of the stream.]  
9  
10

## 11 Exceptions

Exception	Condition
<code>System.ArgumentException</code>	<i>stream</i> does not support reading.
<code>System.ArgumentNullException</code>	<i>stream</i> or <i>encoding</i> is null.
<code>System.ArgumentOutOfRangeException</code>	<i>bufferSize</i> is less than or equal to zero.

12

13

# 1 StreamReader(System.IO.Stream, 2 System.Text.Encoding, System.Boolean) 3 Constructor

```
4 [ILAsm]  
5 public rtspecialname specialname instance void .ctor(class  
6 System.IO.Stream stream, class System.Text.Encoding encoding, bool  
7 detectEncodingFromByteOrderMarks)  
  
8 [C#]  
9 public StreamReader(Stream stream, Encoding encoding, bool  
10 detectEncodingFromByteOrderMarks)
```

## 11 Summary

12 Constructs and initializes a new instance of the `System.IO.StreamReader` class for the  
13 specified stream, with the specified character encoding and byte order mark detection  
14 option.

## 15 Parameters

Parameter	Description
<i>stream</i>	The <code>System.IO.Stream</code> to read.
<i>encoding</i>	A <code>System.Text.Encoding</code> that specifies the character encoding to use.
<i>detectEncodingFromByteOrderMarks</i>	A <code>System.Boolean</code> value that indicates whether the new <code>System.IO.StreamReader</code> is required to look for byte order marks at the beginning of the stream. Specify <code>true</code> to enable detection of byte order marks; otherwise, specify <code>false</code> .

## 16 17 Description

18 This constructor initializes the `System.IO.StreamReader.CurrentEncoding` property  
19 using *encoding*, the `System.IO.StreamReader.BaseStream` property using *stream*, and  
20 the internal buffer to the default size. [*Note:* The default buffer size is implementation  
21 defined.]

22  
23  
24  
25 If requested, this constructor detects the encoding by examining the first three bytes of  
26 *stream*. This constructor automatically recognizes UTF-8, little-endian Unicode, and big-  
27 endian Unicode text if the stream starts with the appropriate byte order marks.

1 Otherwise, the user-provided encoding is used. See the  
2 `System.Text.Encoding.GetPreamble` method for more information.

### 3 Exceptions

Exception	Condition
<b>System.ArgumentException</b>	<i>stream</i> does not support reading.
<b>System.ArgumentNullException</b>	<i>stream</i> or <i>encoding</i> is null.

4

5

# 1 StreamReader(System.IO.Stream)

## 2 Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(class  
5 System.IO.Stream stream)  
  
6 [C#]  
7 public StreamReader(Stream stream)
```

### 8 Summary

9 Constructs and initializes a new instance of the `System.IO.StreamReader` class for the  
10 specified stream.

### 11 Parameters

Parameter	Description
<i>stream</i>	The <code>System.IO.Stream</code> to read.

### 12 13 Description

14 This constructor initializes the `System.IO.StreamReader.CurrentEncoding` property to  
15 `System.Text.UTF8Encoding`, the `System.IO.StreamReader.BaseStream` property using  
16 *stream*, and the internal buffer to the default size. [*Note:* The default buffer size is  
17 implementation dependent.]  
18  
19

### 20 Exceptions

Exception	Condition
<b>System.ArgumentException</b>	<i>stream</i> does not support reading.
<b>System.ArgumentNullException</b>	<i>stream</i> is null.

21

22

# 1 StreamReader(System.IO.Stream, 2 System.Boolean) Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(class  
5 System.IO.Stream stream, bool detectEncodingFromByteOrderMarks)  
  
6 [C#]  
7 public StreamReader(Stream stream, bool detectEncodingFromByteOrderMarks)
```

## 8 Summary

9 Constructs and initializes a new instance of the `System.IO.StreamReader` class for the  
10 specified stream, with the specified byte order mark detection option.

## 11 Parameters

Parameter	Description
<i>stream</i>	The <i>stream</i> to read.
<i>detectEncodingFromByteOrderMarks</i>	A <code>System.Boolean</code> value that indicates whether the new <code>System.IO.StreamReader</code> is required to look for byte order marks at the beginning of the stream. Specify <code>true</code> to enable detection of byte order marks; otherwise, specify <code>false</code> .

## 12 13 Description

14 This constructor initializes the `System.IO.StreamReader.CurrentEncoding` property to  
15 `System.Text.UTF8Encoding`, the `System.IO.StreamReader.BaseStream` property using  
16 *stream*, and the internal buffer to the default size. [*Note:* The default buffer size is  
17 implementation defined.]

18  
19  
20  
21 If requested, the current constructor detects the encoding by examining the first three  
22 bytes of the stream. The constructor automatically recognizes UTF-8, little-endian  
23 Unicode, and big-endian Unicode text if the file starts with the appropriate byte order  
24 marks. Otherwise, UTF-8 encoding is used. For more information, see the  
25 `System.Text.Encoding.GetPreamble` method.

## 26 Exceptions

Exception	Condition
-----------	-----------

<b>System.ArgumentException</b>	<i>stream</i> does not support reading.
<b>System.ArgumentNullException</b>	<i>stream</i> is null.

1

2

# 1 StreamReader(System.IO.Stream, 2 System.Text.Encoding) Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(class  
5 System.IO.Stream stream, class System.Text.Encoding encoding)  
  
6 [C#]  
7 public StreamReader(Stream stream, Encoding encoding)
```

## 8 Summary

9 Constructs and initializes a new instance of the `System.IO.StreamReader` class for the  
10 specified stream with the specified character encoding.

## 11 Parameters

Parameter	Description
<i>stream</i>	The <code>System.IO.Stream</code> to read.
<i>encoding</i>	A <code>System.Text.Encoding</code> that specifies the character encoding to use.

## 12 13 Description

14 This constructor initializes the `System.IO.StreamReader.CurrentEncoding` property  
15 using *encoding*, the `System.IO.StreamReader.BaseStream` property using *stream*, and  
16 the internal buffer to the default size. [*Note:* The default buffer size is implementation  
17 defined.]  
18  
19

## 20 Exceptions

Exception	Condition
<code>System.ArgumentException</code>	<i>stream</i> does not support reading.
<code>System.ArgumentNullException</code>	<i>stream</i> or <i>encoding</i> is null.

21

22

# 1 StreamReader.Close() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual void Close()  
4 [C#]  
5 public override void Close()
```

## 6 Summary

7 Closes the current instance of `System.IO.StreamReader`, releasing any system  
8 resources associated with it.

## 9 Description

10 Following a call to this method, operations on the current instance might raise  
11 exceptions.

12  
13 [*Note:* This version of `System.IO.StreamReader.Close` is equivalent to  
14 `System.IO.StreamReader.Dispose(true)`.

15  
16 This method overrides `System.IO.TextReader.Close`.

17  
18 ]

19

# 1 StreamReader.DiscardBufferedData() Method

```
2 [ILAsm]  
3 .method public hidebysig instance void DiscardBufferedData()  
4 [C#]  
5 public void DiscardBufferedData()
```

## 6 Summary

7 Allows a `System.IO.StreamReader` to discard its buffered data.

## 8 Description

9 *[Note:* This method is useful when reading from a stream after seeking to a new  
10 position. If this method is not called and the internal buffer is not empty, a read attempt  
11 at the new location will first return data that is in the buffer before returning the text at  
12 the current position in the stream.]

13  
14

15

# 1 StreamReader.Dispose(System.Boolean)

## 2 Method

```
3 [ILAsm]  
4 .method family hidebysig virtual void Dispose(bool disposing)  
5 [C#]  
6 protected override void Dispose(bool disposing)
```

### 7 Summary

8 Releases the unmanaged resources used by the `System.IO.StreamReader` and  
9 optionally releases the managed resources.

### 10 Parameters

Parameter	Description
<i>disposing</i>	true to release both managed and unmanaged resources; false to release only unmanaged resources.

### 11 Description

### 12

13 When the *disposing* parameter is `true`, this method releases all resources held by any  
14 managed objects that this `System.IO.StreamReader` references. This method invokes  
15 the `Dispose()` method of each referenced object.

16  
17 [Note: `System.IO.StreamReader.Dispose` can be called multiple times by other objects.  
18 When overriding `System.IO.StreamReader.Dispose(System.Boolean)`, be careful not  
19 to reference objects that have been previously disposed in an earlier call to  
20 `System.IO.StreamReader.Dispose`.

21  
22 This method calls the dispose method of the base class,  
23 `System.IO.TextReader.Dispose(disposing)`.

24 ]  
25

26

# 1 StreamReader.Peek() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual int32 Peek()  
4 [C#]  
5 public override int Peek()
```

## 6 Summary

7 Returns the next character in the underlying stream without advancing the position of  
8 the `System.IO.StreamReader` in the stream.

## 9 Return Value

10 The next character from the character source as a `System.Int32`, or -1 if at the end of  
11 the stream.

## 12 Description

13 [*Note:* This method returns -1 is when the end of the underlying stream is reached  
14 because a Unicode character can contain only values between hexadecimal 0x0000 to  
15 0xFFFF (0 to 65535).  
16

17 This method overrides `System.IO.TextReader.Peek`.

18 ]  
19

## 20 Exceptions

Exception	Condition
<code>System.IO.IOException</code>	An I/O error occurred.

21

22

# 1 StreamReader.Read() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual int32 Read()  
4 [C#]  
5 public override int Read()
```

## 6 Summary

7 Reads the next character from the input stream and advances the character position by  
8 one character.

## 9 Return Value

10 The next character from the character source represented as a `System.Int32`, or -1 if at  
11 the end of the stream.

## 12 Description

13 [*Note:* This method returns -1 when the end of the underlying stream is reached  
14 because a Unicode character can contain only values between hexadecimal 0x0000 to  
15 0xFFFF (0 to 65535).  
16

17 This method overrides `System.IO.TextReader.Read`.

18  
19 ]

## 20 Exceptions

Exception	Condition
<code>System.IO.IOException</code>	An I/O error occurred.

21

22

# 1 StreamReader.Read(System.Char[], 2 System.Int32, System.Int32) Method

```
3 [ILAsm]  
4 .method public hidebysig virtual int32 Read(char[] buffer, int32 index,  
5 int32 count)  
6 [C#]  
7 public override int Read(char[] buffer, int index, int count)
```

## 8 Summary

9 Reads a maximum of *count* characters from the current stream into *buffer*, beginning at  
10 *index*.

## 11 Parameters

Parameter	Description
<i>buffer</i>	A System.Char array. When this method returns, contains the specified character array with the values between <i>index</i> and ( <i>index</i> + <i>count</i> - 1) replaced by the characters read from the current instance.
<i>index</i>	A System.Int32 that specifies the index of <i>buffer</i> at which to begin writing.
<i>count</i>	A System.Int32 that specifies the maximum number of characters to read.

## 12 13 Return Value

14 A System.Int32 containing the number of characters that have been read, or zero if  
15 there are no more characters left to read. Can be less than *count* if the end of the  
16 stream has been reached.

## 17 Description

18 [Note: This method returns after either *count* characters are read, or the end of the file  
19 is reached. System.IO.TextReader.ReadBlock is a blocking version of  
20 System.IO.StreamReader.Read.

21 This method overrides System.IO.TextReader.Read.  
22  
23  
24 ]

## 25 Exceptions

Exception	Condition
<b>System.ArgumentException</b>	<i>buffer.Length - index &lt; count.</i>
<b>System.ArgumentNullException</b>	<i>buffer</i> is null.
<b>System.ArgumentOutOfRangeException</b>	<i>index</i> or <i>count</i> is negative.
<b>System.IO.IOException</b>	An I/O error occurred.  -or-  The current stream is closed.

1

2

# 1 StreamReader.ReadLine() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual string ReadLine()  
4 [C#]  
5 public override string ReadLine()
```

## 6 Summary

7 Reads a line of characters from the current stream and returns the data as a string.

## 8 Return Value

9 A `System.String` containing the next line from the input stream, or `null` if the end of  
10 the input stream is reached.

## 11 Description

12 [*Note:* This method defines a line as a sequence of characters followed by a carriage  
13 return (hexadecimal 0x000d), a line feed (hexadecimal 0x000a), or  
14 `System.Environment.NewLine`. The returned string does not contain the terminating  
15 character(s).

16 This method overrides `System.IO.TextReader.ReadLine`.

17 ]  
18 ]  
19 ]

## 20 Exceptions

Exception	Condition
<b>System.IO.IOException</b>	An I/O error occurred.
<b>System.OutOfMemoryException</b>	There is insufficient memory to allocate a buffer for the returned string.

21

22

# 1 StreamReader.ReadToEnd() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual string ReadToEnd()  
4 [C#]  
5 public override string ReadToEnd()
```

## 6 Summary

7 Reads the stream from the current position to the end of the stream.

## 8 Return Value

9 A `System.Int32` containing the rest of the stream as a string, from the current position  
10 to the end. If the current position is at the end of the stream, returns the empty string  
11 (`""`).

## 12 Description

13 [*Note:* This method overrides `System.IO.TextReader.ReadToEnd`.  
14 ]  
15 ]

## 16 Exceptions

Exception	Condition
<b>System.IO.IOException</b>	An I/O error occurred.
<b>System.OutOfMemoryException</b>	There is insufficient memory to allocate a buffer for the returned string.

17

18

# 1 StreamReader.BaseStream Property

```
2 [ILAsm]  
3 .property class System.IO.Stream BaseStream { public hidebysig virtual  
4 specialname class System.IO.Stream get_BaseStream() }  
5 [C#]  
6 public virtual Stream BaseStream { get; }
```

## 7 Summary

8 Gets the underlying stream.

## 9 Property Value

10 The underlying `System.IO.Stream` which the current `System.IO.StreamReader` instance  
11 is reading.

## 12 Description

13 This property is read-only.

14

# 1 StreamReader.CurrentEncoding Property

```
2 [ILAsm]  
3 .property class System.Text.Encoding CurrentEncoding { public hidebysig  
4 virtual specialname class System.Text.Encoding get_CurrentEncoding() }  
5 [C#]  
6 public virtual Encoding CurrentEncoding { get; }
```

## 7 Summary

8 Gets the current character encoding that the current `System.IO.StreamReader` is using.

## 9 Property Value

10 The current `System.Text.Encoding` used by the current reader.

## 11 Description

12 This property is read-only.

13  
14 The value returned by this property might change after the first call to a  
15 `System.IO.StreamReader.Read` method if encoding auto detection was specified to the  
16 constructor for the current instance.

17