

1 System.IO.StringReader Class

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

6 Assembly Info:

- 7 • *Name:* mscorlib
- 8 • *Public Key:* [00 00 00 00 00 00 00 00 00 04 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.TextReader` that reads from a string.

16 Inherits From: `System.IO.TextReader`

17
18 **Library:** BCL

19
20 **Thread Safety:** All public static members of this type are safe for multithreaded operations.
21 No instance members are guaranteed to be thread safe.

22

1 StringReader(System.String) Constructor

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

6 Summary

7 Constructs and initializes a new instance of the `System.IO.StringReader` class that
8 reads from the specified string.

9 Parameters

Parameter	Description
s	The <code>System.String</code> to be initialized to.

10

11 Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	s is null.

12

13

1 StringReader.Close() Method

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

6 Summary

7 Closes the System.IO.StringReader.

8 Description

9 Following a call to System.IO.StringReader.Close, other System.IO.StringReader
10 methods on the current instance will throw an exception.

11
12 [*Note:* This version of System.IO.StringReader.Close is equivalent to
13 System.IO.StringReader.Dispose(true).

14
15 This method overrides System.IO.Stream.Close.

16
17]

18

1 `StringReader.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 system resources used by the current instance.

9 Parameters

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

10 11 Description

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

15
16 [Note: `System.IO.StringReader.Dispose` can be called multiple times by other objects.
17 When overriding `System.IO.StringReader.Dispose(System.Boolean)`, be careful not
18 to reference objects that have been previously disposed in an earlier call to
19 `System.IO.StringReader.Dispose`.]
20
21

22

1 StringReader.Peek() Method

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

6 Summary

7 Returns the next available character but does not advance the reader's position in the
8 underlying string.

9 Return Value

10 The next character to be read as a `System.Int32`, or -1 if no more characters are
11 available.

12 Description

13 The current position of the `System.IO.StringReader` is not changed by this operation.

14
15 *[Note:* This method returns -1 is when the end of the underlying string is reached
16 because a Unicode character can contain only values between hexadecimal 0x0000 to
17 0xFFFF (0 to 65535).

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

19
20
21]

22 Exceptions

Exception	Condition
<code>System.ObjectDisposedException</code>	The current reader is closed.

23

24

1 `StringReader.Read(System.Char[],` 2 `System.Int32, System.Int32)` Method

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

8 Summary

9 Reads a block of characters from the input string.

10 Parameters

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

11 12 Return Value

13 A `System.Int32` containing the total number of characters read into the buffer, or zero if
14 the end of the underlying string has been reached.

15 Description

16 [Note: This method overrides `System.IO.TextReader.Read`.]
17
18

19 Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	<i>buffer</i> is null.
<code>System.ArgumentException</code>	$(index + count) > buffer.Length$.

System.ArgumentOutOfRangeException	<i>index</i> < 0 - or - <i>count</i> < 0.
System.ObjectDisposedException	The current reader is closed.

1

2

1 StringReader.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 string and advances the character position by
8 one character.

9 Return Value

10 The next character from the underlying string as a `System.Int32`, or -1 if no more
11 characters are available.

12 Description

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

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

17]
18]
19]

20 Exceptions

Exception	Condition
<code>System.ObjectDisposedException</code>	The current reader is closed.

21

22

1 StringReader.ReadLine() Method

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

6 Summary

7 Reads a line from the underlying string.

8 Return Value

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

11 Description

12 A line is defined as a sequence of characters followed by a carriage return (0x000d), a
13 line feed (0x000a), or a carriage return immediately followed by a line feed. The
14 resulting string does not contain the terminating character(s).

15
16 [*Note:* This method overrides `System.IO.TextReader.ReadLine.`]
17
18

19 Exceptions

Exception	Condition
System.ObjectDisposedException	The current reader is closed.
System.OutOfMemoryException	There is insufficient memory to allocate a buffer for the returned string.

20

21

1 StringReader.ReadToEnd() Method

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

6 Summary

7 Returns the underlying string from the current position to the end.

8 Return Value

9 A `System.String` containing the content from the current position to the end of the
10 underlying string.

11 Description

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

15 Exceptions

Exception	Condition
System.ObjectDisposedException	The current reader is closed.
System.OutOfMemoryException	There is insufficient memory to allocate a buffer for the returned string.

16
17