

# System.IO.StreamWriter Class

```
[ILASM]
.class public serializable StreamWriter extends
System.IO.TextWriter

[C#]
public class StreamWriter: TextWriter
```

## Assembly Info:

- *Name:* mscorlib
- *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00]
- *Version:* 1.0.x.x
- *Attributes:*
  - CLSCompliantAttribute(true)

## Implements:

- **System.IDisposable**

## Summary

Implements a **System.IO.Stream** wrapper that writes characters to a stream in a particular encoding.

## Inherits From: System.IO.TextWriter

**Library:** BCL

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

## Description

The **System.IO.StreamWriter** class is designed for character output in a particular **System.Text.Encoding**, whereas subclasses of **System.IO.Stream** are designed for byte input and output.

**System.IO.StreamWriter** defaults to using an instance of **System.Text.UTF8Encoding** unless specified otherwise. This instance of **System.Text.UTF8Encoding** is constructed such that the **System.Text.Encoding.GetPreamble** method returns the Unicode byte order mark written in UTF-8. The preamble of the encoding is added to a stream when you are not appending to an existing stream. This means any text file you create with **System.IO.StreamWriter** has three byte order marks at its beginning. UTF-8 handles all Unicode characters correctly and gives consistent results on localized versions

1 of the operating system.

2

3 [*Note:* By default, **System.IO.StreamWriter** is not thread safe. For a  
4 thread-safe wrapper, see **System.IO.TextWriter.Synchronized.**]

5

# 1 StreamWriter(System.IO.Stream)

## 2 Constructor

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

### 8 Summary

9 Constructs and initializes a new instance of the  
10 **System.IO.StreamWriter** class for the specified stream, using the  
11 default encoding and buffer size.

### 12 Parameters

13  
14

Parameter	Description
<i>stream</i>	The <b>System.IO.Stream</b> to write to.

15

### 16 Description

17 This constructor initializes the **System.IO.StreamWriter.Encoding**  
18 property to a **System.Text.UTF8Encoding** whose  
19 **System.Text.Encoding.GetPreamble** method returns an empty byte  
20 array. For additional information, see  
21 **System.IO.TextWriter.Encoding**. The  
22 **System.IO.StreamWriter.BaseStream** property is initialized using  
23 *stream*.

24

25 [Note: The default buffer size may typically be around 4 KB.]

### 26 Exceptions

27  
28

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

29

30

31

# 1 StreamWriter(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  
6 encoding)  
  
7 [C#]  
8 public StreamWriter(Stream stream, Encoding encoding)
```

## 9 Summary

10 Constructs and initializes a new instance of the  
11 **System.IO.StreamWriter** class for the specified stream, using the  
12 specified encoding and the default buffer size.

## 13 Parameters

14  
15

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

16  
17

## Description

18 This constructor initializes the **System.IO.StreamWriter.Encoding**  
19 property using *encoding*, and the  
20 **System.IO.StreamWriter.BaseStream** property using *stream*. For  
21 additional information, see **System.IO.TextWriter.Encoding**.

22  
23

[Note: The default buffer size may typically be around 4 KB.]

## 24 Exceptions

25  
26

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

27  
28  
29

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

```
4 [ILASM]  
5 public rtspecialname specialname instance void .ctor(class  
6 System.IO.Stream stream, class System.Text.Encoding  
7 encoding, int32 bufferSize)
```

```
8 [C#]  
9 public StreamWriter(Stream stream, Encoding encoding, int  
10 bufferSize)
```

## 11 Summary

12 Constructs and initializes a new instance of the  
13 **System.IO.StreamWriter** class for the specified stream, using the  
14 specified encoding and buffer size.

## 15 Parameters

Parameter	Description
<i>stream</i>	The <b>System.IO.Stream</b> to write to.
<i>encoding</i>	A <b>System.Text.Encoding</b> that specifies the character encoding to use.
<i>bufferSize</i>	A <b>System.Int32</b> that specifies the buffer size.

## 18 Description 19

20 This constructor initializes the **System.IO.StreamWriter.Encoding**  
21 property using *encoding*, and the  
22 **System.IO.StreamWriter.BaseStream** property using *stream*. For  
23 additional information, see **System.IO.TextWriter.Encoding**.

## 24 Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<i>stream</i> or <i>encoding</i> is <b>null</b> .
<b>System.ArgumentOutOfRangeException</b>	<i>bufferSize</i> is negative.
<b>System.ArgumentException</b>	<i>stream</i> does not support writing.

27  
28  
29

# 1 StreamWriter(System.String) Constructor

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

## 7 Summary

8 Constructs and initializes a new instance of the  
9 **System.IO.StreamWriter** class for the specified file on the specified  
10 path, using the default encoding and buffer size.

## 11 Parameters

12  
13

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

14  
15

## 15 Description

16 This constructor initializes the **System.IO.StreamWriter.Encoding**  
17 property to a **System.Text.UTF8Encoding** whose  
18 **System.Text.Encoding.GetPreamble** method returns an empty byte  
19 array. For additional information, see  
20 **System.IO.TextWriter.Encoding**.

21  
22  
23  
24  
25

[*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 may be able to access a physical device.

26  
27  
28  
29  
30

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

The default buffer size may typically be around 4 KB.]

## 31 Exceptions

32  
33

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.UnauthorizedAccessException</b>	Access to <i>path</i> is denied.
<b>System.ArgumentException</b>	<i>path</i> is an empty string ("").
<b>System.ArgumentNullException</b>	<i>path</i> is <b>null</b> .
<b>System.Security.SecurityException</b>	The caller does not have the required permission.

1  
2  
3  
4

**Permissions**

Permission	Description
<b>System.Security.Permissions.FileIOPermission</b>	Requires permission for reading and writing files. See <b>System.Security.Permissions.FileIOPermissionAccess.Read</b> , <b>System.Security.Permissions.FileIOPermissionAccess.Write</b>

5  
6  
7

# 1 StreamWriter(System.String, 2 System.Boolean) Constructor

```
3 [ILASM]  
4 public rtspecialname specialname instance void .ctor(string  
5 path, bool append)  
  
6 [C#]  
7 public StreamWriter(string path, bool append)
```

## 8 Summary

9 Constructs and initializes a new instance of the  
10 **System.IO.StreamWriter** class for the specified file on the specified  
11 path, using the default encoding and buffer size.

## 12 Parameters

13  
14

Parameter	Description
<i>path</i>	A <b>System.String</b> that specifies the complete file path to write to.
<i>append</i>	A <b>System.Boolean</b> value that determines whether data is to be appended to the file. If the file exists and <i>append</i> is <b>false</b> , the file is overwritten. If the file exists and <i>append</i> is <b>true</b> , the data is appended to the file. Otherwise, a new file is created.

15

## 16 Description

17 This constructor initializes the **System.IO.StreamWriter.Encoding**  
18 property to **System.Text.UTF8Encoding** whose  
19 **System.Text.Encoding.GetPreamble** method returns an empty byte  
20 array. For additional information, see  
21 **System.IO.TextWriter.Encoding**.

22

23 If the specified file exists, it can be either overwritten or appended to.  
24 If the file does not exist, this constructor creates a new file.

25

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

30

31 For information on the valid format and characters for path strings,  
32 see **System.IO.Path**.

33

34 The default buffer size may typically be around 4 KB.]

1 **Exceptions**

2

3

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.UnauthorizedAccessException</b>	Access to <i>path</i> is denied.
<b>System.ArgumentException</b>	<i>path</i> is an empty string ("").
<b>System.ArgumentNullException</b>	<i>path</i> is <b>null</b> .
<b>System.Security.SecurityException</b>	The caller does not have the required permission.

4

5

**Permissions**

6

7

Permission	Description
<b>System.Security.Permissions.FileIOPermission</b>	Requires permission for reading and writing files. See <b>System.Security.Permissions.FileIOPermissionAccessRead</b> , <b>System.Security.Permissions.FileIOPermissionAccessWrite</b>

8

9

10

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

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

## 10 Summary

11 Constructs and initializes a new instance of the  
12 **System.IO.StreamWriter** class for the specified file on the specified  
13 path, using the specified encoding and default buffer size.

## 14 Parameters

15  
16

Parameter	Description
<i>path</i>	A <b>System.String</b> that specifies the complete file path to write to.
<i>append</i>	A <b>System.Boolean</b> value that determines whether data is to be appended to the file. If the file exists and <i>append</i> is <b>false</b> , the file is overwritten. If the file exists and <i>append</i> is <b>true</b> , the data is appended to the file. Otherwise, a new file is created.
<i>encoding</i>	A <b>System.Text.Encoding</b> that specifies the character encoding to use.

17

## 18 Description

19 If the specified file exists, it can be either overwritten or appended to.  
20 If the file does not exist, this constructor creates a new file.

21

22 This constructor initializes the **System.IO.StreamWriter.Encoding**  
23 property using *encoding*. For additional information, see  
24 **System.IO.TextWriter.Encoding**.

25

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

30

31 For information on the valid format and characters for path strings,  
32 see **System.IO.Path**.

33

34 The default buffer size may typically be around 4 KB.]

1 **Exceptions**

2

3

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.UnauthorizedAccessException</b>	Access to <i>path</i> is denied.
<b>System.ArgumentException</b>	<i>path</i> is an empty string ("").
<b>System.ArgumentNullException</b>	<i>path</i> is <b>null</b> .
<b>System.Security.SecurityException</b>	The caller does not have the required permission.

4

5 **Permissions**

6

7

Permission	Description
<b>System.Security.Permissions.FileIOPermission</b>	Requires permission for reading and writing files. See <b>System.Security.Permissions.FileIOPermissionAccess.Read</b> , <b>System.Security.Permissions.FileIOPermissionAccess.Write</b>

8

9

10

# StreamWriter(System.String, System.Boolean, System.Text.Encoding, System.Int32) Constructor

```
[ILASM]  
public rtspecialname specialname instance void .ctor(string  
path, bool append, class System.Text.Encoding encoding,  
int32 bufferSize)
```

```
[C#]  
public StreamWriter(string path, bool append, Encoding  
encoding, int bufferSize)
```

## Summary

Constructs and initializes a new instance of the **System.IO.StreamWriter** class for the specified file on the specified path, using the specified encoding and buffer size.

## Parameters

Parameter	Description
<i>path</i>	A <b>System.String</b> that specifies the complete file path to write to.
<i>append</i>	A <b>System.Boolean</b> value that determines whether data is to be appended to the file. If the file exists and <i>append</i> is <b>false</b> , the file is overwritten. If the file exists and <i>append</i> is <b>true</b> , the data is appended to the file. Otherwise, a new file is created.
<i>encoding</i>	A <b>System.Text.Encoding</b> that specifies the character encoding to use.
<i>bufferSize</i>	A <b>System.Int32</b> that specifies the buffer size.

## Description

If the specified file exists, it can be either overwritten or appended to. If the file does not exist, this constructor creates a new file.

This constructor initializes the **System.IO.StreamWriter.Encoding** property using *encoding*. For additional information, see **System.IO.TextWriter.Encoding**.

[*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 may be able to access a physical device.

For information on the valid format and characters for path strings, see **System.IO.Path**.]

1 **Exceptions**

2

3

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.ArgumentException</b>	<i>path</i> is an empty string ("").
<b>System.ArgumentNullException</b>	<i>path</i> or <i>encoding</i> is <b>null</b> .
<b>System.ArgumentOutOfRangeException</b>	<i>bufferSize</i> is negative.
<b>System.Security.SecurityException</b>	The caller does not have the required permission.
<b>System.UnauthorizedAccessException</b>	Access to <i>path</i> is denied.

4

5

**Permissions**

6

7

Permission	Description
<b>System.Security.Permissions.FileIOPermission</b>	Requires permission for reading and writing files. See <b>System.Security.Permissions.FileIOPermissionAccess.Read</b> , <b>System.Security.Permissions.FileIOPermissionAccess.Write</b>

8

9

10

# 1 StreamWriter.Close() Method

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

## 6 Summary

7 Closes the current **System.IO.StreamWriter** and the underlying  
8 stream.

## 9 Description

10 This method calls **System.IO.StreamWriter.Flush**, writing buffered  
11 data to the underlying stream. Following a call to  
12 **System.IO.StreamWriter.Close**, any operations on the current  
13 instance might raise exceptions.

14  
15 [Note: This version of **System.IO.StreamWriter.Close** is equivalent  
16 to **System.IO.StreamWriter.Dispose(true)**.

17  
18 This method overrides **System.IO.Stream.Close.**]

19

# 1 StreamWriter.Dispose(System.Boolean) 2 Method

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

## 8 Summary

9 Releases the unmanaged resources used by the  
10 **System.IO.StreamWriter** and optionally releases the managed  
11 resources.

## 12 Parameters

13  
14

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

15

## 16 Description

17 When the *disposing* parameter is **true**, this method releases all  
18 resources held by any managed objects that this  
19 **System.IO.StreamWriter** references. This method invokes the  
20 **Dispose()** method of each referenced object.

21

22 [Note: **System.IO.StreamWriter.Dispose** may be called multiple  
23 times by other objects. When overriding  
24 **System.IO.StreamWriter.Dispose(System.Boolean)**, be careful  
25 not to reference objects that have been previously disposed in an  
26 earlier call to **System.IO.StreamWriter.Dispose**.

27

28 This method calls the dispose method of the base class,  
29 **System.IO.TextWriter.Dispose** (*disposing*).]

30

# 1 StreamWriter.Finalize() Method

```
2 [ILASM]  
3 .method family hidebysig virtual void Finalize()  
4 [C#]  
5 ~StreamWriter()
```

## 6 Summary

7 Releases resources held by the current instance.

## 8 Description

9 [Note: Application code does not call this method; it is automatically  
10 invoked by during garbage collection unless finalization by the garbage  
11 collector has been disabled. For more information, see  
12 **System.GC.SuppressFinalize**, and **System.Object.Finalize**.  
13

14 This method overrides **System.Object.Finalize**.]  
15

# 1 StreamWriter.Flush() Method

```
2 [ILASM]  
3 .method public hidebysig virtual void Flush()  
4  
5 [C#]  
6 public override void Flush()
```

## 6 Summary

7 Clears all buffers for the current writer and causes any buffered data  
8 to be written to the underlying stream.

## 9 Description

10 [Note: This method overrides **System.IO.TextWriter.Flush.**]

## 11 Exceptions

12  
13

Exception	Condition
System.ObjectDisposedException	The current writer is closed.
System.IO.IOException	An I/O error occurred.

14  
15  
16

# 1 StreamWriter.Write(System.String)

## 2 Method

```
3 [ILASM]  
4 .method public hidebysig virtual void Write(string value)  
5  
6 [C#]  
7 public override void Write(string value)
```

### 7 Summary

8 Writes a string to the stream.

### 9 Parameters

10  
11

Parameter	Description
<i>value</i>	The <b>System.String</b> to write to the stream. If <i>value</i> is <b>null</b> , nothing is written.

12  
13

### 13 Description

14 The specified **System.String** is written to the underlying stream  
15 unless the end of the stream is reached prematurely.

16  
17  
18  
19

If **System.IO.StreamWriter.AutoFlush** is **true**,  
**System.IO.StreamWriter.Flush** is invoked automatically.

20

[*Note:* This method overrides **System.IO.TextWriter.Write**.]

### 21 Exceptions

22  
23

Exception	Condition
<b>System.ObjectDisposedException</b>	<b>System.IO.StreamWriter.AutoFlush</b> is <b>true</b> or the <b>System.IO.StreamWriter</b> buffer is full, and current writer is closed.
<b>System.NotSupportedException</b>	<b>System.IO.StreamWriter.AutoFlush</b> is <b>true</b> or the <b>System.IO.StreamWriter</b> buffer is full, and the contents of the buffer cannot be written to the underlying fixed size stream because the <b>System.IO.StreamWriter</b> is at the end the stream.
<b>System.IO.IOException</b>	An I/O error occurred.

24  
25  
26

# 1 StreamWriter.Write(System.Char[], 2 System.Int32, System.Int32) Method

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

## 9 Summary

10 Writes a sub-array of characters to the underlying stream.

## 11 Parameters

12  
13

Parameter	Description
<i>buffer</i>	A <b>System.Char</b> array containing the data to write.
<i>index</i>	A <b>System.Int32</b> that specifies the index into <i>buffer</i> at which to begin writing.
<i>count</i>	A <b>System.Int32</b> that specifies the number of characters to read from <i>buffer</i> .

14  
15

## Description

16 The specified characters are written to the underlying stream unless  
17 the end of the stream is reached prematurely.

18  
19  
20  
21  
22

If **System.IO.StreamWriter.AutoFlush** is **true**,  
**System.IO.StreamWriter.Flush** is invoked automatically.

[Note: This method overrides **System.IO.TextWriter.Write**.]

## 23 Exceptions

24  
25

Exception	Condition
<b>System.ArgumentNullException</b>	<i>buffer</i> is <b>null</b> .
<b>System.ArgumentException</b>	<i>buffer</i> .Length - <i>index</i> < <i>count</i> .
<b>System.ArgumentOutOfRangeException</b>	<i>index</i> or <i>count</i> is negative.
<b>System.ObjectDisposedException</b>	<b>System.IO.StreamWriter.AutoFlush</b> is <b>true</b> or the <b>System.IO.StreamWriter</b> buffer is full, and current writer is closed.

- 1
- 2
- 3

<b>System.NotSupportedException</b>	<b>System.IO.StreamWriter.AutoFlush</b> is <b>true</b> or the <b>System.IO.StreamWriter</b> buffer is full, and the contents of the buffer cannot be written to the underlying fixed size stream because the <b>System.IO.StreamWriter</b> is at the end the stream.
<b>System.IO.IOException</b>	An I/O error occurred.

# 1 StreamWriter.Write(System.Char[])

## 2 Method

```
3 [ILASM]  
4 .method public hidebysig virtual void Write(class  
5 System.Char[] buffer)  
  
6 [C#]  
7 public override void Write(char[] buffer)
```

### 8 Summary

9 Writes a character array to the underlying stream.

### 10 Parameters

11  
12

Parameter	Description
<i>buffer</i>	A <b>System.Char</b> array containing the data to write. If <i>buffer</i> is <b>null</b> , nothing is written.

13  
14

### 14 Description

15 The specified characters are written to the underlying stream unless  
16 the end of the stream is reached prematurely.

17  
18 If **System.IO.StreamWriter.AutoFlush** is **true**,  
19 **System.IO.StreamWriter.Flush** is invoked automatically.

20  
21 [Note: This method overrides **System.IO.TextWriter.Write**.]

### 22 Exceptions

23  
24

Exception	Condition
<b>System.ObjectDisposedException</b>	<b>System.IO.StreamWriter.AutoFlush</b> is <b>true</b> or the <b>System.IO.StreamWriter</b> buffer is full, and current writer is closed.
<b>System.NotSupportedException</b>	<b>System.IO.StreamWriter.AutoFlush</b> is <b>true</b> or the <b>System.IO.StreamWriter</b> buffer is full, and the contents of the buffer cannot be written to the underlying fixed size stream because the <b>System.IO.StreamWriter</b> is at the end the stream.
<b>System.IO.IOException</b>	An I/O error occurred.

1  
2  
3

# 1 StreamWriter.Write(System.Char) Method

```
2 [ILASM]  
3 .method public hidebysig virtual void Write(valuetype  
4 System.Char value)  
  
5 [C#]  
6 public override void Write(char value)
```

## 7 Summary

8 Writes a character to the stream.

## 9 Parameters

Parameter	Description
<i>value</i>	The <b>System.Char</b> to write to the underlying stream.

## 13 Description

14 The specified character is written to the underlying stream unless the  
15 end of the stream is reached prematurely.

16  
17 If **System.IO.StreamWriter.AutoFlush** is **true**,  
18 **System.IO.StreamWriter.Flush** is invoked automatically.

19  
20 [Note: This method overrides **System.IO.TextWriter.Write**.]

## 21 Exceptions

Exception	Condition
<b>System.ObjectDisposedException</b>	<b>System.IO.StreamWriter.AutoFlush</b> is <b>true</b> or the <b>System.IO.StreamWriter</b> buffer is full, and current writer is closed.
<b>System.NotSupportedException</b>	<b>System.IO.StreamWriter.AutoFlush</b> is <b>true</b> or the <b>System.IO.StreamWriter</b> buffer is full, and the contents of the buffer cannot be written to the underlying fixed size stream because the <b>System.IO.StreamWriter</b> is at the end the stream.
<b>System.IO.IOException</b>	An I/O error occurred.

# 1 StreamWriter.AutoFlush Property

```
2 [ILASM]
3 .property bool AutoFlush { public hidebysig virtual
4 specialname bool get_AutoFlush() public hidebysig virtual
5 specialname void set_AutoFlush(bool value) }
6
7 [C#]
8 public virtual bool AutoFlush { get; set; }
```

## 8 Summary

9 Gets or sets a **System.Boolean** value indicating whether the current  
10 **System.IO.StreamWriter** will flush its buffer to the underlying  
11 stream after every call to **System.IO.StreamWriter.Write**.

## 12 Property Value

14 **true** to force **System.IO.StreamWriter** to flush its buffer; otherwise,  
15 **false**.

## 16 Description

17 The **System.IO.StreamWriter** will do a limited amount of buffering,  
18 both internally and potentially in the encoder from the encoding you  
19 passed in. If **System.IO.StreamWriter.AutoFlush** is set to **false**,  
20 the data will be flushed into the underlying stream only when the  
21 buffer is full, or when **System.IO.StreamWriter.Dispose(true)** or  
22 **System.IO.StreamWriter.Close** is called.

23  
24 Setting **System.IO.StreamWriter.AutoFlush** to **true** forces  
25 **System.IO.StreamWriter** to flush the buffered data out of the  
26 encoder and call **System.IO.StreamWriter.Flush** on the stream  
27 every time **System.IO.StreamWriter.Write** is called.

## 28 Behaviors

29 As described above.

30

# 1 StreamWriter.BaseStream Property

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

## 8 Summary

9 Gets the underlying stream.

## 10 Property Value

11

12 The **System.IO.Stream** the current **System.IO.StreamWriter**  
13 instance is writing to.

## 14 Behaviors

15 As described above.

16

# 1 StreamWriter.Encoding Property

```
2 [ILASM]
3 .property class System.Text.Encoding Encoding { public
4 hidebysig virtual specialname class System.Text.Encoding
5 get_Encoding() }
6
7 [C#]
8 public override Encoding Encoding { get; }
```

## 8 Summary

9 Gets the **System.Text.Encoding** in which the output is written.

## 10 Property Value

11

12 The **System.Text.Encoding** specified in the constructor for the  
13 current instance, or **System.Text.UTF8Encoding** if an encoding was  
14 not specified.

## 15 Description

16 [*Note:* This property overrides the **System.IO.TextWriter.Encoding**  
17 property.]

## 18 Behaviors

19 As described above.

## 20 Usage

21 This property is required in some XML scenarios where a header must  
22 be written containing the encoding used by the  
23 **System.IO.StreamWriter**. This allows XML code to consume an  
24 arbitrary **System.IO.StreamWriter** and generate a correct XML  
25 header.

26