

# System.IO.MemoryStream Class

```
[ILAsm]
.class public serializable MemoryStream extends System.IO.Stream

[C#]
public class MemoryStream: Stream
```

## Assembly Info:

- *Name:* mscorlib
- *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- *Version:* 2.0.x.x
- *Attributes:*
  - CLSCompliantAttribute(true)

## Implements:

- **System.IDisposable**

## Summary

Provides support for creating and using a stream whose backing store is memory.

## Inherits From: System.IO.Stream

**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.MemoryStream` class creates streams that have memory as a backing store instead of a disk or a network connection. `System.IO.MemoryStream` encapsulates data stored as an unsigned byte array. The encapsulated data is directly accessible in memory. Memory streams can reduce the need for temporary buffers and files in an application.

The *current position* of a stream is the position at which the next read or write operation takes place. The current position can be retrieved or set through the `System.IO.MemoryStream.Seek` method. When a new instance of `System.IO.MemoryStream` is created, the current position is set to zero.

The maximum length of a `System.IO.MemoryStream` is implementation-specific.

[*Note:* Memory streams created with an unsigned byte array provide a non-resizable stream view of the data. When using a byte array, you can neither append to nor shrink the stream, although you might be able to modify the existing contents depending on

1 the parameters passed into the constructor.]  
2  
3

4

# MemoryStream(System.Byte[], System.Int32, System.Int32, System.Boolean, System.Boolean) Constructor

```
[ILAsm]
public rtspecialname specialname instance void .ctor(class System.Byte[]
buffer, int32 index, int32 count, bool writable, bool publiclyVisible)

[C#]
public MemoryStream(byte[] buffer, int index, int count, bool writable,
bool publiclyVisible)
```

## Summary

Constructs and initializes a new instance of the `System.IO.MemoryStream` class.

## Parameters

Parameter	Description
<i>buffer</i>	The <code>System.Byte</code> array from which to create the new stream.
<i>index</i>	A <code>System.Int32</code> that specifies the index into <i>buffer</i> at which the stream begins.
<i>count</i>	A <code>System.Int32</code> that specifies the length of the stream in bytes.
<i>writable</i>	A <code>System.Boolean</code> that specifies whether the new stream instance supports writing.
<i>publiclyVisible</i>	A <code>System.Boolean</code> that specifies whether <i>buffer</i> is exposed via <code>System.IO.MemoryStream.GetBuffer</code> , which returns the <code>System.Byte</code> array from which the stream was created. Specify <code>true</code> to expose <i>buffer</i> ; otherwise, specify <code>false</code> .

## Description

The `System.IO.MemoryStream.CanRead` and `System.IO.MemoryStream.CanSeek` properties of the new `System.IO.MemoryStream` instance are set to `true`. The `System.IO.MemoryStream.Capacity` property is set to *count*. The `System.IO.Stream.CanWrite` property is set to *writable*.

[Note: The new stream instance can be written to depending on the value of *writable*, but the `System.IO.MemoryStream.Capacity` of the underlying `System.Byte` array

1 cannot be changed. The length of the stream cannot be set to a value larger than  
2 `System.IO.MemoryStream.Capacity`, but the stream can be truncated (see  
3 `System.IO.MemoryStream.SetLength`).]

4  
5  
6 **Exceptions**

Exception	Condition
<b>System.ArgumentNullException</b>	<i>buffer</i> is null.
<b>System.ArgumentOutOfRangeException</b>	<i>index</i> or <i>count</i> is negative.
<b>System.ArgumentException</b>	( <i>index</i> + <i>count</i> ) is greater than the length of <i>buffer</i> .

7

8

# 1 MemoryStream(System.Byte[], 2 System.Int32, System.Int32, 3 System.Boolean) Constructor

```
4 [ILAsm]  
5 public rtspecialname specialname instance void .ctor(class System.Byte[]  
6 buffer, int32 index, int32 count, bool writable)  
  
7 [C#]  
8 public MemoryStream(byte[] buffer, int index, int count, bool writable)
```

## 9 Summary

10 Constructs and initializes a new non-resizable instance of the System.IO.MemoryStream  
11 class.

## 12 Parameters

Parameter	Description
<i>buffer</i>	The System.Byte array from which to create the new stream.
<i>index</i>	A System.Int32 that specifies the index in <i>buffer</i> at which the stream begins.
<i>count</i>	A System.Int32 that specifies the length of the stream in bytes.
<i>writable</i>	A System.Boolean that specifies whether the new stream instance supports writing.

13

## 14 Description

15 The System.IO.MemoryStream.CanRead and System.IO.MemoryStream.CanSeek  
16 properties of the new System.IO.MemoryStream are set to true. The  
17 System.IO.MemoryStream.Capacity property is set to *count*. The  
18 System.IO.Stream.CanWrite property is set to *writable*.

19

20 [Note: The new stream instance can be written to depending on the value of *writable*,  
21 but the System.IO.MemoryStream.Capacity of the underlying byte array cannot be  
22 changed. The length of the stream cannot be set to a value larger than  
23 System.IO.MemoryStream.Capacity, but the stream can be truncated (see  
24 System.IO.MemoryStream.SetLength).]

25

26

27

28 The new stream does not expose the underlying byte buffer, and calls to the

- 1 `System.IO.MemoryStream.GetBuffer` method throw
- 2 `System.UnauthorizedAccessException`.

3 **Exceptions**

Exception	Condition
<b>System.ArgumentNullException</b>	<i>buffer</i> is null.
<b>System.ArgumentOutOfRangeException</b>	<i>index</i> or <i>count</i> are negative.
<b>System.ArgumentException</b>	( <i>index</i> + <i>count</i> ) is greater than the length of <i>buffer</i> .

4

5

# 1 MemoryStream() Constructor

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

## 6 Summary

7 Constructs and initializes a new resizable instance of the `System.IO.MemoryStream`  
8 class.

## 9 Description

10 The `System.IO.Stream.CanRead`, `System.IO.Stream.CanSeek`, and  
11 `System.IO.Stream.CanWrite` properties of the new instance of the  
12 `System.IO.MemoryStream` class are set to `true`.

13  
14 The capacity of the new stream instance can be increased by using the  
15 `System.IO.MemoryStream.SetLength` method or by setting the  
16 `System.IO.MemoryStream.Capacity` property.

17  
18 The new stream exposes the underlying byte buffer, which can be accessed through the  
19 `System.IO.MemoryStream.GetBuffer` method.

20

# 1 MemoryStream(System.Int32) Constructor

```
2 [ILAsm]  
3 public rtspecialname specialname instance void .ctor(int32 capacity)  
4 [C#]  
5 public MemoryStream(int capacity)
```

## 6 Summary

7 Constructs and initializes a new resizable instance of the `System.IO.MemoryStream`  
8 class.

## 9 Parameters

Parameter	Description
<i>capacity</i>	A <code>System.Int32</code> that specifies the initial size of the internal <code>System.Byte</code> array.

10

## 11 Description

12 The `System.IO.Stream.CanRead`, `System.IO.Stream.CanSeek`, and  
13 `System.IO.Stream.CanWrite` properties of the new instance of the  
14 `System.IO.MemoryStream` class are set to true.

15

16 The `System.IO.MemoryStream.Capacity` of the new stream instance is set to *capacity*  
17 can be increased by using the `System.IO.MemoryStream.SetLength` method or by  
18 setting the `System.IO.MemoryStream.Capacity` property. Write operations at the end of  
19 the new instance of the `System.IO.MemoryStream` class expand the  
20 `System.IO.MemoryStream`.

21

22 The new stream exposes the underlying byte buffer, which can be accessed through the  
23 `System.IO.MemoryStream.GetBuffer` method.

## 24 Exceptions

Exception	Condition
<b>System.ArgumentOutOfRangeException</b>	<i>capacity</i> is negative.

25

26

# 1 MemoryStream(System.Byte[]) Constructor

```
2 [ILAsm]  
3 public rtspecialname specialname instance void .ctor(class System.Byte[]  
4 buffer)  
5 [C#]  
6 public MemoryStream(byte[] buffer)
```

## 7 Summary

8 Constructs and initializes a new non-resizable instance of the `System.IO.MemoryStream`  
9 class.

## 10 Parameters

Parameter	Description
<i>buffer</i>	The <code>System.Byte</code> array from which to create the new stream.

11

## 12 Description

13 The `System.IO.Stream.CanRead`, `System.IO.Stream.CanSeek`, and  
14 `System.IO.Stream.CanWrite` properties of the new instance of the  
15 `System.IO.MemoryStream` class are set to true. `System.IO.MemoryStream.Capacity` is  
16 set to the length of the specified `System.Byte` array.

17

18 [*Note:* The new stream instance can be written to, but the  
19 `System.IO.MemoryStream.Capacity` of the underlying `System.Byte` array cannot be  
20 changed. The length of the stream cannot be set to a value greater than  
21 `System.IO.MemoryStream.Capacity`, but the stream can be truncated (see  
22 `System.IO.MemoryStream.SetLength`).]

23

24

25

26 The new stream does not expose the underlying byte buffer, and calls to the  
27 `System.IO.MemoryStream.GetBuffer` method throw  
28 `System.UnauthorizedAccessException`.

## 29 Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	The <i>buffer</i> parameter is null.

30

31

# 1 MemoryStream(System.Byte[], 2 System.Boolean) Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(class System.Byte[]  
5 buffer, bool writable)  
  
6 [C#]  
7 public MemoryStream(byte[] buffer, bool writable)
```

## 8 Summary

9 Constructs and initializes a new non-resizable instance of the `System.IO.MemoryStream`  
10 class.

## 11 Parameters

Parameter	Description
<i>buffer</i>	The <code>System.Byte</code> array from which to create the new stream.
<i>writable</i>	A <code>System.Boolean</code> that specifies whether the new stream instance supports writing.

12

## 13 Description

14 The `System.IO.Stream.CanRead` and `System.IO.Stream.CanSeek` properties of the new  
15 instance of the `System.IO.MemoryStream` class are set to `true`. The  
16 `System.IO.MemoryStream.Capacity` property is set to the length of the specified  
17 `System.Byte` array. The `System.IO.Stream.CanWrite` property is set to *writable*.

18

19 [*Note:* The new stream instance can be written to depending on the value of *writable*,  
20 but the `System.IO.MemoryStream.Capacity` of the underlying `System.Byte` array  
21 cannot be changed. The length of the stream cannot be set to a value larger than  
22 `System.IO.MemoryStream.Capacity`, but the stream can be truncated (see  
23 `System.IO.MemoryStream.SetLength`).]

24

25

26

27 The new stream does not expose the underlying `System.Byte` buffer, and calls to the  
28 `System.IO.MemoryStream.GetBuffer` method throw  
29 `System.UnauthorizedAccessException`.

## 30 Exceptions

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

**System.ArgumentNullException**

*buffer* is null.

1

2

# 1 MemoryStream(System.Byte[], 2 System.Int32, System.Int32) Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(class System.Byte[]  
5 buffer, int32 index, int32 count)  
  
6 [C#]  
7 public MemoryStream(byte[] buffer, int index, int count)
```

## 8 Summary

9 Constructs and initializes a new non-resizable instance of the System.IO.MemoryStream  
10 class.

## 11 Parameters

Parameter	Description
<i>buffer</i>	The System.Byte array from which to create the new stream.
<i>index</i>	A System.Int32 that specifies the index into <i>buffer</i> at which the stream begins.
<i>count</i>	A System.Int32 that specifies the length of the stream in bytes.

12

## 13 Description

14 The System.IO.Stream.CanRead, System.IO.Stream.CanSeek, and  
15 System.IO.Stream.CanWrite properties of the new System.IO.MemoryStream instance  
16 are set to true. The System.IO.MemoryStream.Capacity property is set to *count*.

17

18 [Note: The new stream instance can be written to, but the  
19 System.IO.MemoryStream.Capacity of the underlying System.Byte array cannot be  
20 changed. The length of the stream cannot be set to a value larger than  
21 System.IO.MemoryStream.Capacity, but the stream can be truncated (see  
22 System.IO.MemoryStream.SetLength).]

23

24

25

26 The new stream does not expose the underlying System.Byte buffer, and calls to the  
27 System.IO.MemoryStream.GetBuffer method throw  
28 System.UnauthorizedAccessException.

## 29 Exceptions

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

<b>System.ArgumentNullException</b>	<i>buffer</i> is null.
<b>System.ArgumentOutOfRangeException</b>	<i>index</i> or <i>count</i> is less than zero.
<b>System.ArgumentException</b>	( <i>index</i> + <i>count</i> ) is greater than the length of <i>buffer</i> .

1

2

# 1 `MemoryStream.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.MemoryStream` instance.

## 8 **Description**

9 The stream will not support reading or writing after this method is invoked. Following a  
10 call to `System.IO.MemoryStream.Close`, operations on the stream can raise an  
11 exception.

12  
13 The buffer of a closed `System.IO.MemoryStream` is still available, and the  
14 `System.IO.MemoryStream.ToArray` and `System.IO.MemoryStream.GetBuffer` methods  
15 can be called successfully.

16  
17 [*Note:* This method overrides `System.IO.Stream.Close`.]

18

# 1 MemoryStream.Flush() Method

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

## 6 Summary

7 Overrides `System.IO.Stream.Flush` so that no action is performed.

## 8 Description

9 Since any data written to a `System.IO.MemoryStream` is written into RAM, this method  
10 is redundant.

11  
12 [*Note:* This method overrides `System.IO.Stream.Flush`.]  
13  
14

15

# 1 MemoryStream.GetBuffer() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual class System.Byte[] GetBuffer()  
4 [C#]  
5 public virtual byte[] GetBuffer()
```

## 6 Summary

7 Returns the array of unsigned bytes from which this stream was created.

## 8 Return Value

9 The `System.Byte` array from which the current stream was created, or the underlying  
10 array if a `System.Byte` array was not provided to the `System.IO.MemoryStream`  
11 constructor during construction of the current instance.

## 12 Description

13 To create a `System.IO.MemoryStream` instance with a publicly visible buffer use the  
14 default constructor, `System.IO.MemoryStream( System.Byte [], System.Int32,`  
15 `System.Int32, System.Boolean, true)` or `System.IO.MemoryStream(System.Int32 )`  
16 constructor.

17  
18 If the current stream is resizable, multiple calls to this method do not return the same  
19 array if the underlying `System.Byte` array is resized between calls. For additional  
20 information, see `System.IO.MemoryStream.Capacity`.

21  
22 [*Note:* This method works when the `System.IO.MemoryStream` is closed.]  
23  
24

## 25 Behaviors

26 As described above.

27

## 28 Exceptions

Exception	Condition
<b>System.UnauthorizedAccessException</b>	The current instance was not created with a publicly visible buffer.

29

## 30 Example

1 The following example demonstrates that two calls to the  
2 System.IO.MemoryStream.GetBuffer method on a resizable stream do not return the  
3 same array if the underlying byte array is reallocated.

```
4  
5 [C#  
  
6 using System;  
7 using System.IO;  
8  
9 public class MemoryStreamTest {  
10     public static void Main() {  
11  
12         MemoryStream ms = new MemoryStream(10);  
13  
14         byte[] a = ms.GetBuffer();  
15         byte[] b = ms.GetBuffer();  
16  
17         //Force reallocation of the underlying byte array.  
18         ms.Capacity = 10240;  
19         byte[] c = ms.GetBuffer();  
20  
21  
22         if(Object.ReferenceEquals(a, b))  
23             Console.WriteLine("a and b represent the same instance.");  
24         else  
25             Console.WriteLine("a and b represent the different instances.");  
26  
27         if(Object.ReferenceEquals(a, c))  
28             Console.WriteLine("a and c represent the same instance.");  
29         else  
30             Console.WriteLine("a and c represent the different instances.");  
31  
32     }  
33 }
```

34 The output is

```
35  
36 a and b represent the same instance.  
37  
38  
39 a and c represent the different instances.  
40
```

41

# 1 MemoryStream.Read(System.Byte[], 2 System.Int32, System.Int32) Method

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

## 8 Summary

9 Reads a block of bytes from the current stream at the current position, and writes the  
10 data to the specified byte array.

## 11 Parameters

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

## 12 13 Return Value

14 A `System.Int32` that specifies the total number of bytes written into the buffer, or zero  
15 if the end of the stream is reached before any bytes are read.

## 16 Description

17 If the read operation is successful, the current position within the stream advances by  
18 the number of bytes read. If an exception occurs, the current position within the stream  
19 remains unchanged.

20  
21 If the read takes place immediately following a seek beyond the end of the stream, the  
22 end of the stream is reached.

23  
24 [Note: If the byte array specified in the *buffer* parameter is the underlying buffer  
25 returned by the `System.IO.MemoryStream.GetBuffer` method, the array contents are  
26 overwritten, and no exception is thrown.

27  
28 This method overrides `System.IO.Stream.Read`.

1  
2 ]

### 3 Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<i>buffer</i> is null.
<b>System.ArgumentOutOfRangeException</b>	<i>offset</i> or <i>count</i> is negative.
<b>System.ArgumentException</b>	( <i>offset</i> + <i>count</i> ) is larger than the length of <i>buffer</i> .
<b>System.ObjectDisposedException</b>	The current stream is closed.

### 4 5 Example

6 The following example demonstrates the result of reading from a  
7 `System.IO.MemoryStream` into its underlying byte array.

8  
9 [C#]

```
10 using System;  
11 using System.IO;  
12  
13 public class MemoryStreamTest {  
14     public static void Main() {  
15  
16         byte[] values = new byte [] {0,1,2,3,4,5,6,7,8,9};  
17  
18         foreach (byte b in values) {  
19             Console.Write(b);  
20         }  
21  
22         Console.WriteLine();  
23  
24         MemoryStream ms = new MemoryStream (values);  
25  
26         ms.Read(values, 1, 5);  
27  
28         foreach (byte b in values) {  
29             Console.Write(b);  
30         }  
31     }  
32 }
```

33 The output is

34  
35 0123456789  
36

1  
2 0012346789  
3  
4

# 1 MemoryStream.ReadByte() Method

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

## 6 Summary

7 Reads a byte from the current stream at the current position.

## 8 Return Value

9 The byte cast to a `System.Int32`, or -1 if the end of the stream has been reached.

## 10 Description

11 If the read operation is successful, the current position within the stream is advanced by  
12 one byte. If an exception occurs, the current position within the stream is unchanged.

13  
14 If the read takes place immediately following a seek beyond the end of the stream, the  
15 end of the stream is reached.

16  
17 [*Note:* This method overrides `System.IO.Stream.ReadByte`.]  
18  
19

## 20 Exceptions

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

21

22

# 1 MemoryStream.Seek(System.Int64, 2 System.IO.SeekOrigin) Method

```
3 [ILAsm]  
4 .method public hidebysig virtual int64 Seek(int64 offset, valuetype  
5 System.IO.SeekOrigin loc)  
  
6 [C#]  
7 public override long Seek(long offset, SeekOrigin loc)
```

## 8 Summary

9 Changes the position within the current stream by the given offset, which is relative to  
10 the stated origin.

## 11 Parameters

Parameter	Description
<i>offset</i>	A <code>System.Int64</code> that specifies the new position within the stream. This is relative to the <i>loc</i> parameter, and can be positive or negative.
<i>loc</i>	A <code>System.IO.SeekOrigin</code> value that specifies the seek reference point.

12

## 13 Return Value

14 A `System.Int64` containing the new position within the stream, calculated by combining  
15 the seek reference point and the offset.

## 16 Description

17 [Note: This method overrides `System.IO.Stream.Seek`.]  
18  
19

## 20 Exceptions

Exception	Condition
<b>System.IO.IOException</b>	Seeking is attempted before the beginning of the stream.
<b>System.ArgumentOutOfRangeException</b>	<i>offset</i> is greater than the maximum length of <code>System.IO.MemoryStream</code> .

<b>System.ArgumentException</b>	<i>loc</i> is not a valid <code>System.IO.SeekOrigin</code> value.
<b>System.ObjectDisposedException</b>	The current stream is closed.

1

2

# 1 MemoryStream.SetLength(System.Int64)

## 2 Method

```
3 [ILAsm]  
4 .method public hidebysig virtual void SetLength(int64 value)  
5 [C#]  
6 public override void SetLength(long value)
```

### 7 Summary

8 Sets the length of the current stream to the specified value.

### 9 Parameters

Parameter	Description
<i>value</i>	A System.Int64 that specifies the value at which to set the length.

10

### 11 Description

12 If the specified value is less than the current length of the stream, the stream is  
13 truncated. If after the truncation the current position within the stream is past the end  
14 of the stream, the System.IO.MemoryStream.ReadByte method returns -1, the  
15 System.IO.MemoryStream.Read method reads zero bytes into the provided byte array,  
16 and System.IO.MemoryStream.Write and System.IO.MemoryStream.WriteByte  
17 methods append specified bytes at the end of the stream, increasing its length.

18

19 If the specified value is larger than the current capacity and the stream is resizable, the  
20 capacity is increased, and the current position within the stream is unchanged. If the  
21 length is increased, the contents of the stream between the old and the new length are  
22 initialized to zeros.

23

24 [*Note:* A System.IO.MemoryStream instance must support writing for this method to  
25 work. Use the System.IO.MemoryStream.CanWrite property to determine whether the  
26 current instance supports writing. For additional information, see  
27 System.IO.Stream.CanWrite.

28

29 This method overrides System.IO.Stream.SetLength.

30

31 ]

### 32 Exceptions

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

<p><b>System.NotSupportedException</b></p>	<p>The current stream is not resizable and <i>value</i> is greater than the current <code>System.IO.MemoryStream.Capacity</code>.</p> <p>-or-</p> <p>The current stream does not support writing.</p>
<p><b>System.ArgumentOutOfRangeException</b></p>	<p><i>value</i> is negative or is greater than the maximum length of the <code>System.IO.MemoryStream - origin</code>, where <i>origin</i> is the index into the underlying buffer at which the stream starts.</p>

1

2

# 1 MemoryStream.ToArray() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual class System.Byte[] ToArray()  
4 [C#]  
5 public virtual byte[] ToArray()
```

## 6 Summary

7 Writes the entire stream contents to a `System.Byte` array, regardless of the current  
8 position within the stream.

## 9 Return Value

10 A new `System.Byte` array.

## 11 Description

12 This method returns a copy of the contents of the `System.IO.MemoryStream` as a byte  
13 array. If the current instance was constructed on a provided byte array, a copy of the  
14 section of the array to which the current instance has access is returned. [*Note:* For  
15 additional information, see the `System.IO.MemoryStream (System.Byte[],  
16 System.Int32, System.Int32 )` constructor.]

17  
18  
19  
20  
21  
22

[*Note:* This method works when the `System.IO.MemoryStream` is closed.]

## 23 Behaviors

24 As described above.

25

26

# 1 MemoryStream.Write(System.Byte[], 2 System.Int32, System.Int32) Method

```
3 [ILAsm]  
4 .method public hidebysig virtual void Write(class System.Byte[] buffer,  
5 int32 offset, int32 count)  
  
6 [C#]  
7 public override void Write(byte[] buffer, int offset, int count)
```

## 8 Summary

9 Writes a block of bytes to the current stream at the current position using data read  
10 from buffer.

## 11 Parameters

Parameter	Description
<i>buffer</i>	The <code>System.Byte</code> array to write data from.
<i>offset</i>	A <code>System.Int32</code> that specifies the zero based byte offset into <i>buffer</i> at which to begin writing from.
<i>count</i>	A <code>System.Int32</code> that specifies the maximum number of bytes to write from <i>buffer</i> .

## 12 13 Description

14 If the write operation is successful, the current position within the stream is advanced  
15 by the number of bytes written. If an exception occurs, the current position within the  
16 stream is unchanged.

17  
18 If the write takes place immediately following a seek beyond the end of the stream, that  
19 stream is zero-byte-extended to the new seek position before the given bytes are  
20 written.

21  
22 Write operations at the end of a resizable `System.IO.MemoryStream` expand the  
23 `System.IO.MemoryStream`.

24  
25 [Note: Use the `System.IO.MemoryStream.CanWrite` method to determine whether the  
26 current stream supports writing.]

27  
28  
29  
30 [Note: This method overrides `System.IO.Stream.Write`.]

1  
2

### 3 Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<i>buffer</i> is null.
<b>System.NotSupportedException</b>	The current stream does not support writing.  -or-  The current position is closer than <i>count</i> bytes to the end of the stream, and the capacity cannot be modified.
<b>System.ArgumentException</b>	( <i>offset</i> + <i>count</i> ) is greater than the length of <i>buffer</i> .
<b>System.ArgumentOutOfRangeException</b>	<i>offset</i> or <i>count</i> are negative.
<b>System.IO.IOException</b>	An I/O error occurred.
<b>System.ObjectDisposedException</b>	The current stream is closed.

4

5

# 1 `MemoryStream.WriteByte(System.Byte)`

## 2 Method

```
3 [ILAsm]  
4 .method public hidebysig virtual void WriteByte(unsigned int8 value)  
5 [C#]  
6 public override void WriteByte(byte value)
```

### 7 Summary

8 Writes a `System.Byte` to the current stream at the current position.

### 9 Parameters

Parameter	Description
<i>value</i>	The <code>System.Byte</code> to write.

### 10 11 Description

12 Write operations at the end of a resizable `System.IO.MemoryStream` expand the  
13 `System.IO.MemoryStream`. If the write operation is successful, the current position  
14 within the stream is advanced by one byte. If an exception occurs, the position is  
15 unchanged.

16  
17 If the write takes place immediately following a seek beyond the end of the stream, that  
18 stream is zero-byte-extended to the new seek position before the given byte is written.

19  
20 [*Note:* Use the `System.IO.MemoryStream.CanWrite` method to determine whether the  
21 current stream supports writing.]

22  
23  
24  
25 [*Note:* This method overrides `System.IO.Stream.WriteByte`.]

### 26 27 28 Exceptions

Exception	Condition
<b>System.ObjectDisposedException</b>	The current stream is closed.
<b>System.NotSupportedException</b>	The current stream does not support writing.

	<p>-or-</p> <p>The current position is at the end of the stream, and the stream's capacity cannot be modified.</p>
--	--

1

2

# 1 MemoryStream.WriteTo(System.IO.Stream) 2 Method

```
3 [ILAsm]  
4 .method public hidebysig virtual void WriteTo(class System.IO.Stream  
5 stream)  
6 [C#]  
7 public virtual void WriteTo(Stream stream)
```

## 8 Summary

9 Writes the entire contents of the current `System.IO.MemoryStream` instance to a  
10 specified stream.

## 11 Parameters

Parameter	Description
<i>stream</i>	The <code>System.IO.Stream</code> to write the current memory stream to.

## 12 13 Description

14 [Note: This method is equivalent to calling `stream.Write(this.GetBuffer(), 0,`  
15 `this.GetBuffer().Length)` and passing in the underlying buffer of the current  
16 instance.]  
17  
18

## 19 Behaviors

20 As described above.  
21

## 22 Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<i>stream</i> is null.
<b>System.ObjectDisposedException</b>	The current or target stream is closed.

23

24

# 1 MemoryStream.CanRead Property

```
2 [ILAsm]  
3 .property bool CanRead { public hidebysig virtual specialname bool  
4 get_CanRead() }  
  
5 [C#]  
6 public override bool CanRead { get; }
```

## 7 Summary

8 Gets a `System.Boolean` value indicating whether the current stream supports reading.

## 9 Property Value

10 `true` if the current stream is open and supports reading; otherwise `false`.

## 11 Description

12 This property is read-only.

13

# 1 MemoryStream.CanSeek Property

```
2 [ILAsm]  
3 .property bool CanSeek { public hidebysig virtual specialname bool  
4 get_CanSeek() }  
  
5 [C#]  
6 public override bool CanSeek { get; }
```

## 7 Summary

8 Gets a `System.Boolean` value indicating whether the current stream supports seeking.

## 9 Property Value

10 `true` if the stream is open and supports seeking; otherwise `false`.

## 11 Description

12 This property is read-only.

13

# 1 MemoryStream.CanWrite Property

```
2 [ILAsm]  
3 .property bool CanWrite { public hidebysig virtual specialname bool  
4 get_CanWrite() }  
  
5 [C#]  
6 public override bool CanWrite { get; }
```

## 7 Summary

8 Gets a `System.Boolean` value indicating whether the current stream supports writing.

## 9 Property Value

10 `true` if the stream supports writing; otherwise, `false`.

## 11 Description

12 This property is read-only.

13

# 1 MemoryStream.Capacity Property

```
2 [ILAsm]  
3 .property int32 Capacity { public hidebysig virtual specialname int32  
4 get_Capacity() public hidebysig virtual specialname void  
5 set_Capacity(int32 value) }  
  
6 [C#]  
7 public virtual int Capacity { get; set; }
```

## 8 Summary

9 Gets or sets the number of bytes allocated for the current stream.

## 10 Property Value

11 A `System.Int32` containing the number of bytes allocated for the current stream.

## 12 Description

13 `System.IO.MemoryStream.Capacity` is the buffer length for system-provided byte  
14 arrays. If the current stream is created with a specified `System.Byte` array,  
15 `System.IO.MemoryStream.Capacity` indicates the length of the portion of the provided  
16 array to which the current stream has access. [*Note:* For additional information, see the  
17 `System.IO.MemoryStream (System.Byte[], System.Int32, System.Int32 )`  
18 constructor.]

19  
20  
21  
22 `System.IO.MemoryStream.Capacity` cannot be set to a value less than the current  
23 length of the stream, but can be set to less than the current capacity. If the capacity  
24 specified is less than the current capacity, the size of the buffer used to hold the stream  
25 can be reduced, but need not be.

26  
27 [*Note:* If the value specified for a set operation is less than the default value, for  
28 performance reasons the property is set to the default. The default value of the  
29 `System.IO.MemoryStream.Capacity` property is unspecified.]  
30  
31

## 32 Behaviors

33 As described above.

## 35 Exceptions

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

<b>System.ArgumentOutOfRangeException</b>	The value specified for a set operation is negative or less than the current length of the stream.
<b>System.NotSupportedException</b>	A set operation was attempted on a stream whose capacity cannot be modified.
<b>System.ObjectDisposedException</b>	The current stream is closed.

1

2

# 1 MemoryStream.Length Property

```
2 [ILAsm]  
3 .property int64 Length { public hidebysig virtual specialname int64  
4 get_Length() }  
5 [C#]  
6 public override long Length { get; }
```

## 7 Summary

8 Gets the length of the stream in bytes.

## 9 Property Value

10 A `System.Int64` containing the length of the stream in bytes.

## 11 Description

12 This property is read-only.

13  
14 [*Note:* This property overrides `System.IO.Stream.Length`.]  
15  
16

## 17 Exceptions

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

18

19

# 1 MemoryStream.Position Property

```
2 [ILAsm]
3 .property int64 Position { public hidebysig virtual specialname int64
4 get_Position() public hidebysig virtual specialname void
5 set_Position(int64 value) }
6 [C#]
7 public override long Position { get; set; }
```

## 8 Summary

9 Gets or sets the current position within the stream.

## 10 Property Value

11 A System.Int64 containing the current position within the stream.

## 12 Description

13 [*Note:* This property overrides System.IO.Stream.Position.]

14

## 16 Exceptions

Exception	Condition
<b>System.ArgumentOutOfRangeException</b>	The value specified for a set operation is negative or greater than the maximum length of a System.IO.MemoryStream.
<b>System.ObjectDisposedException</b>	The current stream is closed.

17

18