

# System.TimeSpan Structure

```
[ILAsm]
.class public sequential sealed serializable TimeSpan extends
System.ValueType implements System.IComparable,
System.IComparable`1<valuetype System.TimeSpan>,
System.IEquatable`1<valuetype System.TimeSpan>

[C#]
public struct TimeSpan: IComparable, IComparable<TimeSpan>,
IEquatable<TimeSpan>
```

## Assembly Info:

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

## Implements:

- **System.IComparable**
- **System.IComparable<System.TimeSpan>**
- **System.IEquatable<System.TimeSpan>**

## Summary

Represents an interval of time.

## Inherits From: System.ValueType

**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.TimeSpan` structure represents an interval of time with values ranging from `System.Int64.MinValue` to `System.Int64.MaxValue` 100-nanosecond *ticks*.

[*Note:* The value of a `System.TimeSpan` is represented internally as a number of 100-nanosecond ticks. Both the specification of a number of ticks and the value of a `System.TimeSpan` can be positive or negative.

A `System.TimeSpan` can be represented as a string in the format "[*-*]d.hh:mm:ss.ff" where "*-*" is an optional sign for negative `System.TimeSpan` values, the "d" component is days, "hh" is hours, "mm" is minutes, "ss" is seconds, and "ff" is fractions of a second.

1 For example, a `System.TimeSpan` initialized with  $10^{13}$  ticks would be represented as  
2 "11.13:46:40", which is 11 days, 13 hours, 46 minutes, and 40 seconds.  
3  
4 Due to a varying number of days in months and years, the longest unit of time that is  
5 used by this structure is the day.  
6  
7 ]  
8

# 1 TimeSpan(System.Int64) Constructor

```
2 [ILAsm]  
3 public rtspecialname specialname instance void .ctor(int64 ticks)  
  
4 [C#]  
5 public TimeSpan(long ticks)
```

## 6 Summary

7 Constructs and initializes a new `System.TimeSpan` with the specified number of ticks.

## 8 Parameters

Parameter	Description
<i>ticks</i>	A <code>System.Int64</code> that specifies the number of ticks with which to initialize the new <code>System.TimeSpan</code> .

# 1 TimeSpan(System.Int32, System.Int32, 2 System.Int32) Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(int32 hours, int32  
5 minutes, int32 seconds)  
  
6 [C#]  
7 public TimeSpan(int hours, int minutes, int seconds)
```

## 8 Summary

9 Constructs and initializes a new `System.TimeSpan` with the specified numbers of hours,  
10 minutes, and seconds.

## 11 Parameters

Parameter	Description
<i>hours</i>	A <code>System.Int32</code> that specifies the number of hours with which to initialize the new <code>System.TimeSpan</code> .
<i>minutes</i>	A <code>System.Int32</code> that specifies the number of minutes with which to initialize the new <code>System.TimeSpan</code> .
<i>seconds</i>	A <code>System.Int32</code> that specifies the number of seconds with which to initialize the new <code>System.TimeSpan</code> .

## 12 Description

13 The specified *hours*, *minutes*, and *seconds* are converted to ticks, and that value is used  
14 to initialize the new `System.TimeSpan`.

## 15 Exceptions

Exception	Condition
<b>System.ArgumentOutOfRangeException</b>	The parameters specify a <code>System.TimeSpan</code> value less than <code>System.TimeSpan.MinValue</code> or greater than <code>System.TimeSpan.MaxValue</code> .

# TimeSpan(System.Int32, System.Int32, System.Int32, System.Int32, System.Int32) Constructor

```
[ILAsm]
public rtspecialname specialname instance void .ctor(int32 days, int32
hours, int32 minutes, int32 seconds, int32 milliseconds)

[C#]
public TimeSpan(int days, int hours, int minutes, int seconds, int
milliseconds)
```

## Summary

Constructs and initializes a new `System.TimeSpan` with the specified numbers of days, hours, minutes, seconds, and milliseconds.

## Parameters

Parameter	Description
<i>days</i>	A <code>System.Int32</code> that specifies the number of days with which to initialize the new <code>System.TimeSpan</code> .
<i>hours</i>	A <code>System.Int32</code> that specifies the number of hours with which to initialize the new <code>System.TimeSpan</code> .
<i>minutes</i>	A <code>System.Int32</code> that specifies the number of minutes with which to initialize the new <code>System.TimeSpan</code> .
<i>seconds</i>	A <code>System.Int32</code> that specifies the number of seconds with which to initialize the new <code>System.TimeSpan</code> .
<i>milliseconds</i>	A <code>System.Int32</code> that specifies the number of milliseconds with which to initialize the new <code>System.TimeSpan</code> .

## Description

The specified *days*, *hours*, *minutes*, *seconds*, and *milliseconds* are converted to ticks, and that value is used to initialize the new `System.TimeSpan`.

## Exceptions

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

**System.ArgumentOutOfRangeException**

The parameters specify a `System.TimeSpan` value less than `System.TimeSpan.MinValue` or greater than `System.TimeSpan.MaxValue`.

1

2

# 1 TimeSpan(System.Int32, System.Int32, 2 System.Int32, System.Int32) Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(int32 days, int32  
5 hours, int32 minutes, int32 seconds)  
  
6 [C#]  
7 public TimeSpan(int days, int hours, int minutes, int seconds)
```

## 8 Summary

9 Constructs and initializes a new `System.TimeSpan` with the specified numbers of days,  
10 hours, minutes, and seconds.

## 11 Parameters

Parameter	Description
<i>days</i>	A <code>System.Int32</code> that specifies the number of days with which to initialize the new <code>System.TimeSpan</code> .
<i>hours</i>	A <code>System.Int32</code> that specifies the number of hours with which to initialize the new <code>System.TimeSpan</code> .
<i>minutes</i>	A <code>System.Int32</code> that specifies the number of minutes with which to initialize the new <code>System.TimeSpan</code> .
<i>seconds</i>	A <code>System.Int32</code> that specifies the number of seconds with which to initialize the new <code>System.TimeSpan</code> .

## 13 Description

14 The specified *days*, *hours*, *minutes*, and *seconds* are converted to ticks, and that value  
15 is used to initialize the new `System.TimeSpan`.

## 16 Exceptions

Exception	Condition
<b>System.ArgumentOutOfRangeException</b>	The parameters specify a <code>System.TimeSpan</code> value less than <code>System.TimeSpan.MinValue</code> or greater than <code>System.TimeSpan.MaxValue</code> .





# 1 TimeSpan.MaxValue Field

```
2 [ILAsm]  
3 .field public static initOnly valuetype System.TimeSpan MaxValue  
  
4 [C#]  
5 public static readonly TimeSpan MaxValue
```

## 6 Summary

7 Returns a System.TimeSpan whose value is the maximum value for the  
8 System.TimeSpan type.

## 9 Description

10 This field is read-only.

11  
12 This field is a System.TimeSpan containing System.Int64.MaxValue ticks, the maximum  
13 System.TimeSpan value. The string representation of this value is positive  
14 10675199.02:48:05.4775807.

# 1 TimeSpan.MinValue Field

```
2 [ILAsm]  
3 .field public static initOnly valuetype System.TimeSpan MinValue  
  
4 [C#]  
5 public static readonly TimeSpan MinValue
```

## 6 Summary

7 Returns a `System.TimeSpan` whose value is the minimum value for the  
8 `System.TimeSpan` type.

## 9 Description

10 This field is read-only.

11  
12 This field is a `System.TimeSpan` containing `System.Int64.MinValue` ticks, the minimum  
13 `System.TimeSpan` value. The string representation of this value is negative  
14 10675199.02:48:05.4775808.

# 1 TimeSpan.TicksPerDay Field

```
2 [ILAsm]  
3 .field public static literal int64 TicksPerDay = 864000000000  
4 [C#]  
5 public const long TicksPerDay = 864000000000
```

## 6 Summary

7 Represents the number of ticks in 1 day.

## 8 Description

9 The value of this constant is 864 billion ( $8.64 \times 10^{11}$  ).

# 1 TimeSpan.TicksPerHour Field

```
2 [ILAsm]  
3 .field public static literal int64 TicksPerHour = 36000000000  
4 [C#]  
5 public const long TicksPerHour = 36000000000
```

## 6 Summary

7 Represents the number of ticks in 1 hour.

## 8 Description

9 The value of this constant is 36 billion ( $3.6 \times 10^{10}$  ).

10

# 1 TimeSpan.TicksPerMillisecond Field

```
2 [ILAsm]  
3 .field public static literal int64 TicksPerMillisecond = 10000  
  
4 [C#]  
5 public const long TicksPerMillisecond = 10000
```

## 6 Summary

7 Represents the number of ticks in 1 millisecond.

## 8 Description

9 The value of this constant is 10 thousand ( $10^4$  ).

10

# 1 TimeSpan.TicksPerMinute Field

```
2 [ILAsm]  
3 .field public static literal int64 TicksPerMinute = 600000000  
4 [C#]  
5 public const long TicksPerMinute = 600000000
```

## 6 Summary

7 Represents the number of ticks in 1 minute.

## 8 Description

9 The value of this constant is 600 million ( $6 \times 10^8$  ).

10

# 1 TimeSpan.TicksPerSecond Field

```
2 [ILAsm]  
3 .field public static literal int64 TicksPerSecond = 10000000  
4 [C#]  
5 public const long TicksPerSecond = 10000000
```

## 6 Summary

7 Represents the number of ticks in 1 second.

## 8 Description

9 The value of this constant is 10 million ( $10^7$  ).

## 1 TimeSpan.Zero Field

```
2 [ILAsm]
3 .field public static initOnly valuetype System.TimeSpan Zero
4
5 [C#]
6 public static readonly TimeSpan Zero
```

## 6 Summary

7 Returns a `System.TimeSpan` whose value is 0.

## 8 Description

9 This field is read-only.

10  
11 This field is a `System.TimeSpan` whose value is 0 ticks. [*Note:* This provides a  
12 convenient source for 0 in `System.TimeSpan` calculations.]



# 1 TimeSpan.Add(System.TimeSpan) Method

```
2 [ILAsm]
3 .method public hidebysig instance valuetype System.TimeSpan Add(valuetype
4 System.TimeSpan ts)

5 [C#]
6 public TimeSpan Add(TimeSpan ts)
```

## 7 Summary

8 Adds the specified `System.TimeSpan` to the current instance.

## 9 Parameters

Parameter	Description
<i>ts</i>	A <code>System.TimeSpan</code> instance to add to the current instance.

## 11 Return Value

12 A `System.TimeSpan` that represents the value of the current instance plus the value of  
13 *ts*.

## 14 Exceptions

Exception	Condition
<b>System.OverflowException</b>	The sum of the value of the current instance and the value of <i>ts</i> is less than <code>System.TimeSpan.MinValue</code> or greater than <code>System.TimeSpan.MaxValue</code> .

## 16 Example

17 This example demonstrates the `System.TimeSpan.Add` method.

18 [C#]

```
20 using System;
21 public class TimeSpanAddExample {
22     public static void Main() {
23         TimeSpan ts = new TimeSpan(Int32.MaxValue);
24         Console.WriteLine("The value of the timespan 'ts' is {0}", ts);
25         Console.WriteLine("ts.Add(ts) = {0}", ts.Add(ts));
26     }
27 }
```

```
1 The output is
2
3 The value of the timespan 'ts' is 00:03:34.7483647
4
5
6 ts.Add(ts) = 00:07:09.4967294
7
8
```

# 1 TimeSpan.Compare(System.TimeSpan, 2 System.TimeSpan) Method

```
3 [ILAsm]  
4 .method public hidebysig static int32 Compare(valuetype System.TimeSpan  
5 t1, valuetype System.TimeSpan t2)
```

```
6 [C#]  
7 public static int Compare(TimeSpan t1, TimeSpan t2)
```

## 8 Summary

9 Returns the sort order of two System.TimeSpan structures.

## 10 Parameters

Parameter	Description
<i>t1</i>	The first System.TimeSpan to compare.
<i>t2</i>	The second System.TimeSpan to compare.

## 12 Return Value

13 The return value is a negative number, zero, or a positive number reflecting the sort  
14 order of *t1* as compared to *t2*. For non-zero return values, the exact value returned by  
15 this method is unspecified. The following table defines the return value:

Value	Condition
Any negative number	$t1 < t2$ .
Zero	$t1 == t2$ .
Any positive number	$t1 > t2$ .

# 1 TimeSpan.CompareTo(System.Object)

## 2 Method

```
3 [ILAsm]  
4 .method public final hidebysig virtual int32 CompareTo(object value)  
  
5 [C#]  
6 public int CompareTo(object value)
```

## 7 Summary

8 Returns the sort order of the current instance compared to the specified `System.Object`.

## 9 Parameters

Parameter	Description
<i>value</i>	The <code>System.Object</code> to compare to the current instance.

## 11 Return Value

12 The return value is a negative number, zero, or a positive number reflecting the sort  
13 order of the current instance as compared to *value*. For non-zero return values, the  
14 exact value returned by this method is unspecified. The following table defines the  
15 return value:

Value	Condition
Any negative number	Current instance < <i>value</i> .
Zero	Current instance == <i>value</i> .
Any positive number	Current instance > <i>value</i> , or <i>value</i> is a null reference.

## 17 Description

18 [Note: This method is implemented to support the `System.IComparable` interface.]  
19  
20

## 21 Exceptions

Exception	Condition
<b>System.ArgumentException</b>	<i>value</i> is not a <code>System.TimeSpan</code> and is not a null reference.

1

2

# 1 TimeSpan.CompareTo(System.TimeSpan)

## 2 Method

```
3 [ILAsm]  
4 .method public final hidebysig virtual int32 CompareTo(valuetype  
5 System.TimeSpan value)  
  
6 [C#]  
7 public int CompareTo(TimeSpan value)
```

### 8 Summary

9 Returns the sort order of the current instance compared to the specified  
10 System.TimeSpan.

### 11 Parameters

Parameter	Description
<i>value</i>	The System.TimeSpan to compare to the current instance.

### 13 Return Value

14 The return value is a negative number, zero, or a positive number reflecting the sort  
15 order of the current instance as compared to *value*. For non-zero return values, the  
16 exact value returned by this method is unspecified. The following table defines the  
17 return value:

Value	Condition
Any negative number	Current instance < <i>value</i> .
Zero	Current instance == <i>value</i> .
Any positive number	Current instance > <i>value</i> .

### 19 Description

20 [Note: This method is implemented to support the  
21 System.IComparable<System.TimeSpan> interface.]  
22  
23

# 1 TimeSpan.Duration() Method

```
2 [ILAsm]
3 .method public hidebysig instance valuetype System.TimeSpan Duration()
4 [C#]
5 public TimeSpan Duration()
```

## 6 Summary

7 Returns a `System.TimeSpan` whose value is the absolute value of the current instance.

## 8 Return Value

9 A `System.TimeSpan` whose value is the absolute value of the current instance.

## 10 Exceptions

Exception	Condition
<b>System.OverflowException</b>	The value of the current instance is <code>System.TimeSpan.MinValue</code> .

## 12 Example

13 The following example demonstrates the `System.TimeSpan.Duration` method.

```
14 [C#]
15
16 using System;
17 public class TimeSpanDurationExample {
18     public static void Main() {
19         TimeSpan ts = new TimeSpan(Int32.MinValue);
20         Console.Write("The absolute value of TimeSpan {0} ", ts);
21         Console.WriteLine("is {0}", ts.Duration());
22     }
23 }
```

24 The output is

25  
26 The absolute value of TimeSpan -00:03:34.7483648 is 00:03:34.7483648

# 1    TimeSpan.Equals(System.TimeSpan, 2    System.TimeSpan) Method

```
3    [ILAsm]  
4    .method public hidebysig static bool Equals(valuetype System.TimeSpan t1,  
5    valuetype System.TimeSpan t2)  
  
6    [C#]  
7    public static bool Equals(TimeSpan t1, TimeSpan t2)
```

## 8    Summary

9       Determines whether two System.TimeSpan structures represent the same type and  
10      value.

## 11   Parameters

Parameter	Description
<i>t1</i>	The first instance of System.TimeSpan to compare for equality.
<i>t2</i>	The second instance of System.TimeSpan to compare for equality.

## 13   Return Value

14      true if *t1* and *t2* represent the same value; otherwise, false.



# 1 TimeSpan.Equals(System.Object) Method

```
2 [ILAsm]  
3 .method public hidebysig virtual bool Equals(object value)  
  
4 [C#]  
5 public override bool Equals(object value)
```

## 6 Summary

7 Determines whether the current instance and the specified `System.Object` represent the  
8 same type and value.

## 9 Parameters

Parameter	Description
<i>value</i>	The <code>System.Object</code> to compare to the current instance.

## 11 Return Value

12 `true` if *value* represents the same type and value as the current instance. If *value* is a  
13 null reference or is not a `System.TimeSpan`, returns `false`.

## 14 Description

15 [Note: This method overrides `System.Object.Equals`.]  
16  
17  
18

# 1 TimeSpan.Equals(System.TimeSpan) Method

```
2 [ILAsm]  
3 .method public hidebysig virtual bool Equals(valuetype System.TimeSpan  
4 obj)  
  
5 [C#]  
6 public override bool Equals(TimeSpan obj)
```

## 7 Summary

8 Determines whether the current instance and the specified System.TimeSpan represent  
9 the same value.

## 10 Parameters

Parameter	Description
<i>value</i>	The System.TimeSpan to compare to the current instance.

## 12 Return Value

13 true if *value* represents the same value as the current instance; otherwise, false.

## 14 Description

15 [Note: This method is implemented to support the  
16 System.IEquatable<System.TimeSpan> interface.]  
17  
18  
19

**The following member must be implemented if the ExtendedNumerics library is present in the implementation.**

## TimeSpan.FromDays(System.Double) Method

```
[ILAsm]  
.method public hidebysig static valuetype System.TimeSpan FromDays(float64  
value)  
  
[C#]  
public static TimeSpan FromDays(double value)
```

### Summary

Returns a `System.TimeSpan` that represents the specified number of days where the specification is accurate to the nearest millisecond.

### Parameters

Parameter	Description
<i>value</i>	A <code>System.Double</code> that specifies the number of days with which the new <code>System.TimeSpan</code> is initialized.

### Return Value

A `System.TimeSpan` that represents *value*.

### Description

*value* will only be considered accurate to the nearest millisecond.

If *value* is `System.Double.PositiveInfinity`, a `System.TimeSpan` with the value `System.TimeSpan.MaxValue` is returned. If *value* is `System.Double.NegativeInfinity`, a `System.TimeSpan` with the value `System.TimeSpan.MinValue` is returned.

### Exceptions

Exception	Condition
<b>System.OverflowException</b>	The <code>System.TimeSpan</code> represented by <i>value</i> is greater than <code>System.TimeSpan.MaxValue</code> or less than <code>System.TimeSpan.MinValue</code> .
<b>System.ArgumentException</b>	<i>value</i> is equal to <code>System.Double.NaN</code> .

**The following member must be implemented if the ExtendedNumerics library is present in the implementation.**

## TimeSpan.FromHours(System.Double) Method

```
[ILAsm]  
.method public hidebysig static valuetype System.TimeSpan  
FromHours(float64 value)  
  
[C#]  
public static TimeSpan FromHours(double value)
```

### Summary

Returns a `System.TimeSpan` that represents the specified number of hours where the specification is accurate to the nearest millisecond.

### Parameters

Parameter	Description
<i>value</i>	A <code>System.Double</code> that specifies the number of hours with which the new <code>System.TimeSpan</code> is initialized.

### Return Value

A `System.TimeSpan` that represents *value*.

### Description

*value* will only be considered accurate to the nearest millisecond.

If *value* is `System.Double.PositiveInfinity`, a `System.TimeSpan` with the value `System.TimeSpan.MaxValue` is returned. If *value* is `System.Double.NegativeInfinity`, a `System.TimeSpan` with the value `System.TimeSpan.MinValue` is returned.

### Exceptions

Exception	Condition
<b>System.OverflowException</b>	The <code>System.TimeSpan</code> represented by <i>value</i> is greater than <code>System.TimeSpan.MaxValue</code> or less than <code>System.TimeSpan.MinValue</code> .
<b>System.ArgumentException</b>	<i>value</i> is equal to <code>System.Double.NaN</code> .

1

2

**The following member must be implemented if the ExtendedNumerics library is present in the implementation.**

## TimeSpan.FromMilliseconds(System.Double) Method

```
[ILAsm]  
.method public hidebysig static valuetype System.TimeSpan  
FromMilliseconds(float64 value)  
  
[C#]  
public static TimeSpan FromMilliseconds(double value)
```

### Summary

Returns a `System.TimeSpan` that represents the specified number of milliseconds where the specification is accurate to the nearest millisecond.

### Parameters

Parameter	Description
<i>value</i>	A <code>System.Double</code> that specifies the number of milliseconds with which the new <code>System.TimeSpan</code> is initialized.

### Return Value

A `System.TimeSpan` that represents *value*.

### Description

*value* will only be considered accurate to the nearest millisecond.

If *value* is `System.Double.PositiveInfinity`, a `System.TimeSpan` with the value `System.TimeSpan.MaxValue` is returned. If *value* is `System.Double.NegativeInfinity`, a `System.TimeSpan` with the value `System.TimeSpan.MinValue` is returned.

### Exceptions

Exception	Condition
<b>System.OverflowException</b>	The <code>System.TimeSpan</code> represented by <i>value</i> is greater than <code>System.TimeSpan.MaxValue</code> or less than <code>System.TimeSpan.MinValue</code> .
<b>System.ArgumentException</b>	<i>value</i> is equal to <code>System.Double.NaN</code> .

1

2

**The following member must be implemented if the ExtendedNumerics library is present in the implementation.**

## TimeSpan.FromMinutes(System.Double) Method

```
[ILAsm]  
.method public hidebysig static valuetype System.TimeSpan  
FromMinutes(float64 value)  
  
[C#]  
public static TimeSpan FromMinutes(double value)
```

### Summary

Returns a `System.TimeSpan` that represents the specified number of minutes where the specification is accurate to the nearest millisecond.

### Parameters

Parameter	Description
<i>value</i>	A <code>System.Double</code> that specifies the number of minutes with which the new <code>System.TimeSpan</code> is initialized.

### Return Value

A `System.TimeSpan` that represents *value*.

### Description

*value* will only be considered accurate to the nearest millisecond.

If *value* is `System.Double.PositiveInfinity`, a `System.TimeSpan` with the value `System.TimeSpan.MaxValue` is returned. If *value* is `System.Double.NegativeInfinity`, a `System.TimeSpan` with the value `System.TimeSpan.MinValue` is returned.

### Exceptions

Exception	Condition
<b>System.OverflowException</b>	The <code>System.TimeSpan</code> represented by <i>value</i> is greater than <code>System.TimeSpan.MaxValue</code> or less than <code>System.TimeSpan.MinValue</code> .
<b>System.ArgumentException</b>	<i>value</i> is equal to <code>System.Double.NaN</code> .



1

2

**The following member must be implemented if the ExtendedNumerics library is present in the implementation.**

## TimeSpan.FromSeconds(System.Double) Method

```
[ILAsm]  
.method public hidebysig static valuetype System.TimeSpan  
FromSeconds(float64 value)  
  
[C#]  
public static TimeSpan FromSeconds(double value)
```

### Summary

Returns a `System.TimeSpan` that represents the specified number of seconds where the specification is accurate to the nearest millisecond.

### Parameters

Parameter	Description
<i>value</i>	A <code>System.Double</code> that specifies the number of seconds with which the new <code>System.TimeSpan</code> is initialized.

### Return Value

A `System.TimeSpan` that represents *value*.

### Description

*value* will only be considered accurate to the nearest millisecond.

If *value* is `System.Double.PositiveInfinity`, a `System.TimeSpan` with the value `System.TimeSpan.MaxValue` is returned. If *value* is `System.Double.NegativeInfinity`, a `System.TimeSpan` with the value `System.TimeSpan.MinValue` is returned.

### Exceptions

Exception	Condition
<b>System.OverflowException</b>	The <code>System.TimeSpan</code> represented by <i>value</i> is greater than <code>System.TimeSpan.MaxValue</code> or less than <code>System.TimeSpan.MinValue</code> .
<b>System.ArgumentException</b>	<i>value</i> is equal to <code>System.Double.NaN</code> .

1

2

# 1 TimeSpan.FromTicks(System.Int64) Method

```
2 [ILAsm]  
3 .method public hidebysig static valuetype System.TimeSpan FromTicks(int64  
4 value)  
  
5 [C#]  
6 public static TimeSpan FromTicks(long value)
```

## 7 Summary

8 Returns a System.TimeSpan that represents the specified number of ticks.

## 9 Parameters

Parameter	Description
<i>value</i>	A System.Int64 that specifies the number of ticks with which the new System.TimeSpan is initialized.

## 11 Return Value

12 A System.TimeSpan with a value of *value*.

## 13 Description

14 This method is equivalent to the System.TimeSpan(System.Int64) constructor.

# 1 TimeSpan.GetHashCode() Method

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

## 6 Summary

7 Generates a hash code for the current instance.

## 8 Return Value

9 A `System.Int32` value containing a hash code for the current instance.

## 10 Description

11 The algorithm used to generate the hash code is unspecified.

12  
13 [*Note:* This method overrides `System.Object.GetHashCode()`.]  
14  
15

# 1 TimeSpan.Negate() Method

```
2 [ILAsm]  
3 .method public hidebysig instance valuetype System.TimeSpan Negate()  
  
4 [C#]  
5 public TimeSpan Negate()
```

## 6 Summary

7 Returns a `System.TimeSpan` with the same absolute value but opposite sign as the  
8 current instance.

## 9 Return Value

10 A `System.TimeSpan` with the same absolute value but with the opposite sign as the  
11 current instance.

## 12 Exceptions

Exception	Condition
<b>System.OverflowException</b>	The value of the current instance is <code>System.TimeSpan.MinValue</code> .

# 1 TimeSpan.op\_Addition(System.TimeSpan, System.TimeSpan) Method

```
3 [ILAsm]
4 .method public hidebysig static specialname valuetype System.TimeSpan
5 op_Addition(valuetype System.TimeSpan t1, valuetype System.TimeSpan t2)
6
7 [C#]
8 public static TimeSpan operator +(TimeSpan t1, TimeSpan t2)
```

## 8 Summary

9 Adds the values of two System.TimeSpan instances.

## 10 Parameters

Parameter	Description
<i>t1</i>	The first System.TimeSpan.
<i>t2</i>	The second System.TimeSpan.

## 12 Return Value

13 A System.TimeSpan whose value is the sum of the values of *t1* and *t2*.

## 14 Exceptions

Exception	Condition
<b>System.OverflowException</b>	The sum of <i>t1</i> and <i>t2</i> is less than System.TimeSpan.MinValue or greater than System.TimeSpan.MaxValue.

## 1    TimeSpan.op\_Equality(System.TimeSpan, 2    System.TimeSpan) Method

```
3    [ILAsm]  
4    .method public hidebysig static specialname bool op_Equality(valuetype  
5    System.TimeSpan t1, valuetype System.TimeSpan t2)  
  
6    [C#]  
7    public static bool operator ==(TimeSpan t1, TimeSpan t2)
```

### 8    Summary

9       Determines whether the value of one System.TimeSpan is equal to the value of another  
10      System.TimeSpan.

### 11   Parameters

Parameter	Description
<i>t1</i>	The first System.TimeSpan
<i>t2</i>	The second System.TimeSpan

### 12   Return Value

14      true if the values of *t1* and *t2* are equal; otherwise, false.



## 1    TimeSpan.op\_GreaterThan(System.TimeSpan, 2    System.TimeSpan) Method 3

```
4    [ILAsm]  
5    .method public hidebysig static specialname bool op_GreaterThan(valuetype  
6    System.TimeSpan t1, valuetype System.TimeSpan t2)  
  
7    [C#]  
8    public static bool operator >(TimeSpan t1, TimeSpan t2)
```

### 9    Summary

10    Determines whether the value one `System.TimeSpan` is greater than the value of  
11    another `System.TimeSpan`.

### 12    Parameters

Parameter	Description
<i>t1</i>	The first <code>System.TimeSpan</code> .
<i>t2</i>	The second <code>System.TimeSpan</code> .

### 13    Return Value 14

15    `true` if the value of *t1* is greater than the value of *t2*; otherwise, `false`.  
16

## 1    TimeSpan.op\_GreaterThanOrEqual(System.TimeSpan, System.TimeSpan) Method

```
4    [ILAsm]
5    .method public hidebysig static specialname bool
6    op_GreaterThanOrEqual(valuetype System.TimeSpan t1, valuetype
7    System.TimeSpan t2)

8    [C#]
9    public static bool operator >=(TimeSpan t1, TimeSpan t2)
```

### 10    Summary

11    Determines whether the value of one `System.TimeSpan` is greater than or equal to the  
12    value of another `System.TimeSpan`.

### 13    Parameters

Parameter	Description
<i>t1</i>	The first <code>System.TimeSpan</code> .
<i>t2</i>	The second <code>System.TimeSpan</code> .

### 15    Return Value

16    `true` if the value of *t1* is greater than or equal to the value of *t2*; otherwise, `false`.

# 1 TimeSpan.op\_Inequality(System.TimeSpan, 2 System.TimeSpan) Method

```
3 [ILAsm]  
4 .method public hidebysig static specialname bool op_Inequality(valuetype  
5 System.TimeSpan t1, valuetype System.TimeSpan t2)  
  
6 [C#]  
7 public static bool operator !=(TimeSpan t1, TimeSpan t2)
```

## 8 Summary

9 Determines whether the value of one System.TimeSpan is unequal to the value of  
10 another System.TimeSpan.

## 11 Parameters

Parameter	Description
<i>t1</i>	The first System.TimeSpan.
<i>t2</i>	The second System.TimeSpan.

## 12 Return Value

13 true if the values of *t1* and *t2* are unequal; otherwise, false.  
14  
15

# 1 TimeSpan.op\_LessThan(System.TimeSpan, 2 System.TimeSpan) Method

```
3 [ILAsm]  
4 .method public hidebysig static specialname bool op_LessThan(valuetype  
5 System.TimeSpan t1, valuetype System.TimeSpan t2)  
  
6 [C#]  
7 public static bool operator <(TimeSpan t1, TimeSpan t2)
```

## 8 Summary

9 Determines whether the value of one `System.TimeSpan` is less than the value of another  
10 `System.TimeSpan`.

## 11 Parameters

Parameter	Description
<i>t1</i>	The first <code>System.TimeSpan</code> .
<i>t2</i>	The second <code>System.TimeSpan</code> .

## 12 Return Value

13 `true` if the value of *t1* is less than the value of *t2*; otherwise, `false`.  
14  
15

## 1    TimeSpan.op\_LessThanOrEqual(System.TimeSpan, System.TimeSpan) Method

```
4    [ILAsm]
5    .method public hidebysig static specialname bool
6    op_LessThanOrEqual(valuetype System.TimeSpan t1, valuetype System.TimeSpan
7    t2)

8    [C#]
9    public static bool operator <=(TimeSpan t1, TimeSpan t2)
```

### 10   Summary

11     Determines whether the value of one `System.TimeSpan` is less than or equal to the  
12     value of another `System.TimeSpan`.

### 13   Parameters

Parameter	Description
<i>t1</i>	The first <code>System.TimeSpan</code> .
<i>t2</i>	The second <code>System.TimeSpan</code> .

### 15   Return Value

16     `true` if the value of *t1* is less than or equal to the value of *t2*; otherwise, `false`.

# 1 TimeSpan.op\_Subtraction(System.TimeSpan, System.TimeSpan) Method

```
3 [ILAsm]
4 .method public hidebysig static specialname valuetype System.TimeSpan
5 op_Subtraction(valuetype System.TimeSpan t1, valuetype System.TimeSpan t2)
6
7 [C#]
8 public static TimeSpan operator -(TimeSpan t1, TimeSpan t2)
```

## 8 Summary

9 Subtracts the value of one System.TimeSpan from the value of another  
10 System.TimeSpan.

## 11 Parameters

Parameter	Description
<i>t1</i>	The first System.TimeSpan.
<i>t2</i>	The second System.TimeSpan.

## 13 Return Value

14 A System.TimeSpan whose value is the result of the value of *t1* minus the value of *t2*.

## 15 Exceptions

Exception	Condition
<b>System.OverflowException</b>	The value of <i>t2</i> subtracted from <i>t1</i> is less than System.TimeSpan.MinValue or greater than System.TimeSpan.MaxValue.

## 1    TimeSpan.op\_UnaryNegation(System.TimeSpan) Method

```
4    [ILAsm]
5    .method public hidebysig static specialname valuetype System.TimeSpan
6    op_UnaryNegation(valuetype System.TimeSpan t)
7
7    [C#]
8    public static TimeSpan operator -(TimeSpan t)
```

### 9    Summary

10    Returns a System.TimeSpan whose value is the negated value of a specified  
11    System.TimeSpan.

### 12   Parameters

Parameter	Description
<i>t</i>	A System.TimeSpan whose value will be negated.

### 14   Return Value

15    A System.TimeSpan with the same absolute value but the opposite sign as *t*.

### 16   Exceptions

Exception	Condition
<b>System.OverflowException</b>	<i>t</i> equals System.TimeSpan.MinValue.

# 1 TimeSpan.op\_UnaryPlus(System.TimeSpan)

## 2 Method

```
3 [ILAsm]  
4 .method public hidebysig static specialname valuetype System.TimeSpan  
5 op_UnaryPlus(valuetype System.TimeSpan t)  
  
6 [C#]  
7 public static TimeSpan operator +(TimeSpan t)
```

## 8 Summary

9 Returns the specified instance of System.TimeSpan.

## 10 Parameters

Parameter	Description
<i>t</i>	A System.TimeSpan.

## 12 Return Value

13 System.TimeSpan*t*.

## 14 Description

15 This method returns System.TimeSpan*t*.



# 1 TimeSpan.Parse(System.String) Method

```
2 [ILAsm]  
3 .method public hidebysig static valuetype System.TimeSpan Parse(string s)  
4 [C#]  
5 public static TimeSpan Parse(string s)
```

## 6 Summary

7 Returns the specified `System.String` converted to a `System.TimeSpan` value.

## 8 Parameters

Parameter	Description
<code>s</code>	<p>A <code>System.String</code> containing the value to convert. <code>s</code> contains a time interval in the following form:</p> <p><code>[ws][-][d.]hh:mm:ss[.ff][ws]</code></p> <p>Items in square brackets ('[' and ']') are optional. Colons and periods (':' and '.') are literal characters. For details on the remaining symbols, see the description section.</p>

## 10 Return Value

11 The `System.TimeSpan` value obtained from `s`.

## 12 Description

13 The symbols used in the parameter description for `s` are as follows:

Item	Description
<code>ws</code>	White space (zero or more space and/or tab characters).
<code>"-"</code>	Minus sign, indicating a negative time interval.
<code>"d"</code>	Days.
<code>"hh"</code>	Hours, ranging from 0 to 23 inclusive.
<code>"mm"</code>	Minutes, ranging from 0 to 59 inclusive.

"ss"	Seconds, ranging from 0 to 59 inclusive.
"ff"	Fractional seconds, from 1 to 7 decimal digits inclusive.

1

## 2 Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	s is a null reference.
<b>System.FormatException</b>	s is in an invalid format.
<b>System.OverflowException</b>	s represents a number greater than <code>System.TimeSpan.MaxValue</code> or less than <code>System.TimeSpan.MinValue</code> .  -or-  At least one of the hours, minutes, or seconds components is outside its valid range.

3

## 4 Example

5 This example demonstrates parsing a string to obtain a `System.TimeSpan`.

6

7 [C#]

```

8 using System;
9 public class TimeSpanParseExample {
10     public static void Main() {
11         String str = "    -5.12:34:56.789    ";
12         TimeSpan ts = TimeSpan.Parse(str);
13         Console.WriteLine(@"The string "{0}"", str);
14         Console.WriteLine("pares to TimeSpan {0}", ts);
15     }
16 }

```

18 The output is

19

```

20 The string "    -5.12:34:56.789    "
21 parses to TimeSpan -5.12:34:56.7890000
22
23

```

# 1 TimeSpan.Subtract(System.TimeSpan)

## 2 Method

```
3 [ILAsm]  
4 .method public hidebysig instance valuetype System.TimeSpan  
5 Subtract(valuetype System.TimeSpan ts)  
  
6 [C#]  
7 public TimeSpan Subtract(TimeSpan ts)
```

## 8 Summary

9 Subtracts the value of the specified System.TimeSpan from the value of the current  
10 instance.

## 11 Parameters

Parameter	Description
<i>ts</i>	A System.TimeSpan whose value to subtract from the value of the current instance.

## 13 Return Value

14 A System.TimeSpan whose value is equal to the value of the current instance minus the  
15 value of *ts*.

## 16 Exceptions

Exception	Condition
<b>System.OverflowException</b>	The difference between the value of the current instance and <i>ts</i> is less than System.TimeSpan.MinValue or greater than System.TimeSpan.MaxValue.

# 1 TimeSpan.ToString() Method

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

## 6 Summary

7 Returns a `System.String` representation of the value of the current instance.

## 8 Return Value

9 A `System.String` representation of the current instance formatted as follows:

10 [-][d.]hh:mm:ss[.ff]

11  
12 Items in square brackets ('[' and ']') are included provisionally: '-' is included if and only  
13 if the current instance is negative; "d." and ".ff" are included if and only if those  
14 components are non-zero. Colons and periods (':' and '.') are literal characters. Other  
15 components are as follows.  
16

Component	Description
"-"	Minus sign, indicating a negative time interval.
"d"	Days.
"hh"	Hours, ranging from 0 to 23 inclusive.
"mm"	Minutes, ranging from 0 to 59 inclusive.
"ss"	Seconds, ranging from 0 to 59 inclusive.
"ff"	Fractional seconds.

## 18 Description

19 [Note: This method overrides `System.Object.ToString()`.]  
20  
21

## 22 Example

```
1      This example demonstrates the System.TimeSpan.ToString method.
2
3      [C#]

4      using System;
5      public class TimeSpanToStringExample {
6          public static void Main() {
7              TimeSpan tsOne = new TimeSpan(1, 23, 45, 54, 321);
8              TimeSpan tsTwo = new TimeSpan(0, 23, 45, 54, 0);
9              Console.Write("TimeSpan one, with d. and.ff: ");
10             Console.WriteLine("{0}", tsOne.ToString());
11             Console.Write("TimeSpan two, without d. and.ff: ");
12             Console.WriteLine("{0}", tsTwo.ToString());
13         }
14     }
15     The output is
16
17     TimeSpan one, with d. and.ff: 1.23:45:54.3210000
18
19
20     TimeSpan two, without d. and.ff: 23:45:54
21
22
```

# 1 TimeSpan.Days Property

```
2 [ILAsm]  
3 .property int32 Days { public hidebysig specialname instance int32  
4 get_Days() }  
  
5 [C#]  
6 public int Days { get; }
```

## 7 Summary

8 Gets the number days represented by the current instance.

## 9 Property Value

10 A System.Int32 represents the days component of the current instance. [Note: See  
11 System.TimeSpan.ToString for a more detailed description of the days component.]  
12  
13

## 14 Description

15 This property is read-only.

## 16 Example

17 This example demonstrates using the System.TimeSpan.Days property.  
18

19 [C#]

```
20 using System;  
21 public class TimeSpanPropertiesExampleOne {  
22     public static void Main() {  
23         TimeSpan ts = new TimeSpan((Int64)10e12+3456789);  
24         Console.WriteLine(ts.ToString());  
25         Console.WriteLine("Days: {0}", ts.Days );  
26     }  
27 }
```

28 The output is

29  
30 11.13:46:40.3456789  
31

32  
33 Days: 11  
34  
35

# 1 TimeSpan.Hours Property

```
2 [ILAsm]  
3 .property int32 Hours { public hidebysig specialname instance int32  
4 get_Hours() }  
  
5 [C#]  
6 public int Hours { get; }
```

## 7 Summary

8 Gets the number of hours represented by the current instance.

## 9 Property Value

10 A System.Int32 between 0 and 23 inclusive, that represents the hours component of  
11 the current instance. [*Note:* See System.TimeSpan.ToString for a more detailed  
12 description of the hours component.]  
13  
14

## 15 Description

16 This property is read-only.

## 17 Example

18 This example demonstrates using the System.TimeSpan.Hours property.

```
19 [C#]  
20  
21 using System;  
22 public class TimeSpanPropertiesExampleOne {  
23     public static void Main() {  
24         TimeSpan ts = new TimeSpan((Int64)10e12+3456789);  
25         Console.WriteLine(ts.ToString());  
26         Console.WriteLine("Hours: {0}", ts.Hours );  
27     }  
28 }
```

29 The output is

30  
31 11.13:46:40.3456789

32  
33  
34 Hours: 13  
35  
36

# 1 TimeSpan.Milliseconds Property

```
2 [ILAsm]  
3 .property int32 Milliseconds { public hidebysig specialname instance int32  
4 get_Milliseconds() }  
  
5 [C#]  
6 public int Milliseconds { get; }
```

## 7 Summary

8 Gets the number of milliseconds represented by the current instance.

## 9 Property Value

10 A System.Int32 between 0 and 999 inclusive, that represents the fractional seconds  
11 component of the current instance converted to milliseconds. [Note: See  
12 System.TimeSpan.ToString for a more detailed description of the fractional seconds  
13 component.]  
14  
15

## 16 Description

17 This property is read-only.

## 18 Example

19 This example demonstrates using the System.TimeSpan.Milliseconds property.  
20

21 [C#]

```
22 using System;  
23 public class TimeSpanPropertiesExampleOne {  
24     public static void Main() {  
25         TimeSpan ts = new TimeSpan((Int64)10e12+3456789);  
26         Console.WriteLine(ts.ToString());  
27         Console.WriteLine("Milliseconds: {0}", ts.Milliseconds );  
28     }  
29 }
```

30 The output is

31  
32 11.13:46:40.3456789  
33

34  
35 Milliseconds: 345  
36



# 1 TimeSpan.Minutes Property

```
2 [ILAsm]  
3 .property int32 Minutes { public hidebysig specialname instance int32  
4 get_Minutes() }  
  
5 [C#]  
6 public int Minutes { get; }
```

## 7 Summary

8 Gets the number of minutes represented by the current instance.

## 9 Property Value

10 A System.Int32 between 0 and 59 inclusive, that represents the minutes component of  
11 the current instance. [*Note:* See System.TimeSpan.ToString for a more detailed  
12 description of the minutes component.]  
13  
14

## 15 Description

16 This property is read-only.

## 17 Example

18 This example demonstrates using the System.TimeSpan.Minutes property.

```
19 [C#]  
20  
21 using System;  
22 public class TimeSpanPropertiesExampleOne {  
23     public static void Main() {  
24         TimeSpan ts = new TimeSpan((Int64)10e12+3456789);  
25         Console.WriteLine(ts.ToString());  
26         Console.WriteLine("Minutes: {0}", ts.Minutes );  
27     }  
28 }
```

29 The output is

30  
31 11.13:46:40.3456789  
32

33  
34 Minutes: 46  
35  
36

# 1 TimeSpan.Seconds Property

```
2 [ILAsm]  
3 .property int32 Seconds { public hidebysig specialname instance int32  
4 get_Seconds() }  
  
5 [C#]  
6 public int Seconds { get; }
```

## 7 Summary

8 Gets the number of seconds represented by the current instance.

## 9 Property Value

10 A System.Int32 between 0 and 59 inclusive, that represents the seconds component of  
11 the current instance. [*Note:* See System.TimeSpan.ToString for a more detailed  
12 description of the seconds component.]  
13  
14

## 15 Description

16 This property is read-only.

## 17 Example

18 This example demonstrates using the System.TimeSpan.Seconds property.

```
19 [C#]  
20  
21 using System;  
22 public class TimeSpanPropertiesExampleOne {  
23     public static void Main() {  
24         TimeSpan ts = new TimeSpan((Int64)10e12+3456789);  
25         Console.WriteLine(ts.ToString());  
26         Console.WriteLine("Seconds: {0}", ts.Seconds );  
27     }  
28 }
```

29 The output is

30  
31 11.13:46:40.3456789  
32

33  
34 Seconds: 40  
35

# 1 TimeSpan.Ticks Property

```
2 [ILAsm]  
3 .property int64 Ticks { public hidebysig specialname instance int64  
4 get_Ticks() }  
  
5 [C#]  
6 public long Ticks { get; }
```

## 7 Summary

8 Gets the number of ticks represented by the current instance.

## 9 Property Value

10 A System.Int64 specifying the number of ticks represented by the current instance.

## 11 Description

12 This property is read-only.

**The following member must be implemented if the ExtendedNumerics library is present in the implementation.**

## TimeSpan.TotalDays Property

```
[ILAsm]
.property float64 TotalDays { public hidebysig specialname instance
float64 get_TotalDays() }

[C#]
public double TotalDays { get; }
```

### Summary

Gets the value of the current instance expressed in days.

### Property Value

A `System.Double` that specifies the total number of days represented by the current instance.

### Description

This property is read-only.

[*Note:* This property converts the value of the current instance from ticks to days. This number can include whole and fractional days.]

### Example

This example demonstrates using the `System.TimeSpan.TotalDays` property.

```
[C#]

using System;
public class TimeSpanTotalUnitsProperties{
    public static void Main() {
        TimeSpan ts = new TimeSpan((Int64)10e12);
        Console.WriteLine(ts.ToString());
        Console.WriteLine("TotalDays: {0}", ts.TotalDays);
    }
}
```

The output is

11.13:46:40

TotalDays: 11.5740740740741



**The following member must be implemented if the ExtendedNumerics library is present in the implementation.**

## TimeSpan.TotalHours Property

```
[ILAsm]
.property float64 TotalHours { public hidebysig specialname instance
float64 get_TotalHours() }

[C#]
public double TotalHours { get; }
```

### Summary

Gets the value of the current instance expressed in hours.

### Property Value

A `System.Double` that specifies the total number of hours represented by the current instance.

### Description

This property is read-only.

[*Note:* This property converts the value of the current instance from ticks to hours. This number can include whole and fractional hours.]

### Example

This example demonstrates using the `System.TimeSpan.TotalHours` property.

```
[C#]

using System;
public class TimeSpanTotalUnitsProperties{
    public static void Main() {
        TimeSpan ts = new TimeSpan((Int64)10e12);
        Console.WriteLine(ts.ToString());
        Console.WriteLine("TotalHours: {0}", ts.TotalHours);
    }
}
```

The output is

11.13:46:40

TotalHours: 277.777777777778



**The following member must be implemented if the ExtendedNumerics library is present in the implementation.**

## TimeSpan.TotalMilliseconds Property

```
[ILAsm]
.property float64 TotalMilliseconds { public hidebysig specialname
instance float64 get_TotalMilliseconds() }

[C#]
public double TotalMilliseconds { get; }
```

### Summary

Gets the value of the current instance expressed in milliseconds.

### Property Value

A `System.Double` that specifies the total number of milliseconds represented by the current instance.

### Description

This property is read-only.

[*Note:* This property converts the value of the current instance from ticks to milliseconds. This number can include whole and fractional milliseconds.]

### Example

This example demonstrates using the `System.TimeSpan.TotalMilliseconds` property.

```
[C#]

using System;
public class TimeSpanTotalUnitsProperties{
    public static void Main() {
        TimeSpan ts = new TimeSpan((Int64)10e12);
        Console.WriteLine(ts.ToString());
        Console.WriteLine("TotalMilliseconds: {0}", ts.TotalMilliseconds);
    }
}
```

The output is

11.13:46:40

TotalMilliseconds: 10000000000





**The following member must be implemented if the ExtendedNumerics library is present in the implementation.**

## TimeSpan.TotalMinutes Property

```
[ILAsm]
.property float64 TotalMinutes { public hidebysig specialname instance
float64 get_TotalMinutes() }

[C#]
public double TotalMinutes { get; }
```

### Summary

Gets the value of the current instance expressed in minutes.

### Property Value

A System.Double that specifies the total number of minutes represented by the current instance.

### Description

This property is read-only.

[*Note:* This property converts the value of the current instance from ticks to minutes. This number can include whole and fractional minutes.]

### Example

This example demonstrates using the System.TimeSpan.TotalMinutes property.

```
[C#]

using System;
public class TimeSpanTotalUnitsProperties{
    public static void Main() {
        TimeSpan ts = new TimeSpan((Int64)10e12);
        Console.WriteLine(ts.ToString());
        Console.WriteLine("TotalMinutes: {0}", ts.TotalMinutes);
    }
}
```

The output is

11.13:46:40

TotalMinutes: 16666.6666666667



**The following member must be implemented if the ExtendedNumerics library is present in the implementation.**

## TimeSpan.TotalSeconds Property

```
[ILAsm]
.property float64 TotalSeconds { public hidebysig specialname instance
float64 get_TotalSeconds() }

[C#]
public double TotalSeconds { get; }
```

### Summary

Gets the value of the current instance expressed in seconds.

### Property Value

A `System.Double` that specifies the total number of seconds represented by the current instance.

### Description

This property is read-only.

[*Note:* This property converts the value of the current instance from ticks to seconds. This number can include whole and fractional seconds.]

### Example

This example demonstrates using the `System.TimeSpan.TotalSeconds` property.

```
[C#]

using System;
public class TimeSpanTotalUnitsProperties{
    public static void Main() {
        TimeSpan ts = new TimeSpan((Int64)10e12);
        Console.WriteLine(ts.ToString());
        Console.WriteLine("TotalSeconds: {0}", ts.TotalSeconds);
    }
}
```

The output is

11.13:46:40

TotalSeconds:1000000

