

System.DateTime Structure

```
[ILASM]
.class public sealed serializable DateTime extends
System.ValueType implements System.IComparable,
System.IFormattable

[C#]
public struct DateTime: IComparable, IFormattable
```

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.IComparable
- System.IFormattable

Summary

Represents an instant in time, expressed as a date and time of day.

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.DateTime** value type represents dates and times with values ranging from 00:00:00, 1/1/0001 (Common Era) to 23:59:59 PM, 12/31/9999.

[Note: Time values are measured in 100-nanosecond units, *ticks*, and a particular date is the number of ticks since 12:00 Midnight, January 1, 1 in the Gregorian calendar. For example, a ticks value of 312413760000000000L represents the date, Friday, January 01, 0100 12:00:00 AM.

Time values can be added to, or subtracted from, an instance of **System.DateTime**. Time values can be negative or positive, and

1 expressed in units such as ticks, seconds, or instances of
2 **System.TimeSpan**. Methods and properties in this value type take
3 into account details such as leap years and the number of days in a
4 month.

5
6 12:00:00 AM is Midnight.]

7

1 DateTime(System.Int64) Constructor

```
2 [ILASM]  
3 public rtspecialname specialname instance void .ctor(int64  
4 ticks)  
5  
6 [C#]  
7 public DateTime(long ticks)
```

7 Summary

8 Constructs and initializes a new instance of the **System.DateTime**
9 structure with the date and time expressed in 100-nanosecond units.

10 Parameters

Parameter	Description
<i>ticks</i>	A System.Int64 containing the date and time expressed in 100-nanosecond units.

14 Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	The date and time represented by <i>ticks</i> is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue .

1 DateTime(System.Int32, System.Int32, 2 System.Int32) Constructor

```
3 [ILASM]  
4 public rtspecialname specialname instance void .ctor(int32  
5 year, int32 month, int32 day)  
  
6 [C#]  
7 public DateTime(int year, int month, int day)
```

8 Summary

9 Constructs and initializes a new instance of the **System.DateTime**
10 structure with a specified year, month, and day.

11 Parameters

12
13

Parameter	Description
<i>year</i>	A System.Int32 containing the year (1 through 9999).
<i>month</i>	A System.Int32 containing the month (1 through 12).
<i>day</i>	A System.Int32 containing the day (1 through the number of days in <i>month</i>).

14
15

Description

16 The time of day for the resulting **System.DateTime** is midnight
17 (00:00:00).

18 Exceptions

19
20

Exception	Condition
System.ArgumentOutOfRangeException	<i>year</i> is less than 1 or greater than 9999
	-or-
	<i>month</i> is less than 1 or greater than 12
	-or-
	<i>day</i> is less than 1 or greater than the number of days in <i>month</i>

21
22
23

1 DateTime(System.Int32, System.Int32, 2 System.Int32, System.Int32, 3 System.Int32, System.Int32) Constructor

```
4 [ILASM]  
5 public rtspecialname specialname instance void .ctor(int32  
6 year, int32 month, int32 day, int32 hour, int32 minute,  
7 int32 second)  
  
8 [C#]  
9 public DateTime(int year, int month, int day, int hour, int  
10 minute, int second)
```

11 Summary

12 Constructs and initializes a new instance of the **System.DateTime**
13 structure with a specified year, month, day, hour, minute, and second.

14 Parameters

15
16

Parameter	Description
<i>year</i>	A System.Int32 containing the year (1 through 9999).
<i>month</i>	A System.Int32 containing the month (1 through 12).
<i>day</i>	A System.Int32 containing the day (1 through the number of days in <i>month</i>).
<i>hour</i>	A System.Int32 containing the hours (0 through 23).
<i>minute</i>	A System.Int32 containing the minutes (0 through 59).
<i>second</i>	A System.Int32 containing the seconds (0 through 59).

17
18
19
20

Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	<i>year</i> is less than 1 or greater than 9999 -or- <i>month</i> is less than 1 or greater than 12 -or- <i>day</i> is less than 1 or greater than the number of days in <i>month</i> -or-

- 1
- 2
- 3

	<p><i>hour</i> is less than 0 or greater than 23</p> <p>-or-</p> <p><i>minute</i> is less than 0 or greater than 59</p> <p>-or-</p> <p><i>second</i> is less than 0 or greater than 59</p>
--	--

1 DateTime(System.Int32, System.Int32, 2 System.Int32, System.Int32, 3 System.Int32, System.Int32, 4 System.Int32) Constructor

```
5 [ILASM]  
6 public rtspecialname specialname instance void .ctor(int32  
7 year, int32 month, int32 day, int32 hour, int32 minute,  
8 int32 second, int32 millisecond)  
9  
10 [C#]  
11 public DateTime(int year, int month, int day, int hour, int  
minute, int second, int millisecond)
```

12 Summary

13 Constructs and initializes a new instance of the **System.DateTime**
14 structure with a specified year, month, day, hour, minute, second, and
15 millisecond.

16 Parameters

17
18

Parameter	Description
<i>year</i>	A System.Int32 containing the year (1 through 9999).
<i>month</i>	A System.Int32 containing the month (1 through 12).
<i>day</i>	A System.Int32 containing the day (1 through the number of days in <i>month</i>).
<i>hour</i>	A System.Int32 containing the hours (0 through 23).
<i>minute</i>	A System.Int32 containing the minutes (0 through 59).
<i>second</i>	A System.Int32 containing the seconds (0 through 59).
<i>millisecond</i>	A System.Int32 containing the milliseconds.

19
20
21
22

Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	<i>year</i> is less than 1 or greater than 9999 -or- <i>month</i> is less than 1 or greater than 12 -or-

	<p><i>day</i> is less than 1 or greater than the number of days in <i>month</i></p> <p>-or-</p> <p><i>hour</i> is less than 0 or greater than 23</p> <p>-or-</p> <p><i>minute</i> is less than 0 or greater than 59</p> <p>-or-</p> <p><i>second</i> is less than 0 or greater than 59</p> <p>-or-</p> <p><i>millisecond</i> is less than 0 or greater than 999</p>
System.ArgumentException	<p>The specified parameters evaluate to a date less than System.DateTime.MinValue or greater than System.DateTime.MaxValue.</p>

- 1
- 2
- 3

1 DateTime.MaxValue Field

```
2 [ILASM]  
3 .field public static initOnly valuetype System.DateTime  
4 MaxValue  
5 [C#]  
6 public static readonly DateTime MaxValue
```

7 Summary

8 A constant representing the largest possible value of
9 **System.DateTime**.

10 Description

11 This field is read-only.

12
13 The value of this field is equivalent to 23:59:59.9999999, 12/31/9999,
14 exactly one 100-nanosecond tick before 00:00:00, 01/01/10000.

15

1 DateTime.MinValue Field

```
2 [ILASM]  
3 .field public static initOnly valuetype System.DateTime  
4 MinValue  
5 [C#]  
6 public static readonly DateTime MinValue
```

7 Summary

8 A constant representing the smallest possible value of
9 **System.DateTime**.

10 Description

11 This field is read-only.

12
13 The value of this field is equivalent to 00:00:00.0000000, 1/1/0001.

14

1 DateTime.Add(System.TimeSpan) Method

```
2 [ILASM]  
3 .method public hidebysig instance valuetype System.DateTime  
4 Add(valuetype System.TimeSpan value)  
5 [C#]  
6 public DateTime Add(TimeSpan value)
```

7 Summary

8 Adds the value of a specified **System.TimeSpan** instance to the
9 current instance.

10 Parameters

Parameter	Description
<i>value</i>	A System.TimeSpan instance.

14 Return Value

16 A **System.DateTime** instance set to the sum of the date and time of
17 the current instance and the time interval represented by *value*.

18 Description

19 A specified **System.TimeSpan** is added to the current instance of
20 **System.DateTime**, and the result is returned as a new
21 **System.DateTime**. The returned value is equivalent to
22 **System.DateTime**(*value*.Ticks).

23 Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	The resulting System.DateTime is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue .

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

3 DateTime.AddDays(System.Double) 4 Method

```
5 [ILASM]  
6 .method public hidebysig instance valuetype System.DateTime  
7 AddDays(float64 value)  
8 [C#]  
9 public DateTime AddDays(double value)
```

10 Summary

11 Adds a specified number of days to the value of the current instance.

12 Parameters

Parameter	Description
<i>value</i>	A System.Double containing the number of whole and fractional days. For example, 4.5 is equivalent to 4 days, 12 hours, 0 minutes, 0 seconds, 0 milliseconds, and 0 ticks. <i>value</i> can be negative or positive.

16 Return Value

18 A **System.DateTime** instance set to the sum of the date and time
19 represented by the current instance and the number of days
20 represented by *value*.

21 Description

22 [*Note: value is rounded to the nearest tick.*]

23 Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	The resulting System.DateTime is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue .

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

3 DateTime.AddHours(System.Double) 4 Method

```
5 [ILASM]  
6 .method public hidebysig instance valuetype System.DateTime  
7 AddHours(float64 value)  
8 [C#]  
9 public DateTime AddHours(double value)
```

10 Summary

11 Adds a specified number of hours to the value of the current instance.

12 Parameters

Parameter	Description
<i>value</i>	A System.Double containing the number of whole and fractional hours. For example, 4.5 is equivalent to 4 hours, 30 minutes, 0 seconds, 0 milliseconds, and 0 ticks. <i>value</i> can be negative or positive.

16 Return Value

18 A **System.DateTime** instance set to the sum of the date and time
19 represented by the current instance and the number of hours
20 represented by *value*.

21 Description

22 [*Note: value is rounded to the nearest tick.*]

23 Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	The resulting System.DateTime is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue .

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

3 DateTime.AddMilliseconds(System.Double 4) Method

```
5 [ILASM]  
6 .method public hidebysig instance valuetype System.DateTime  
7 AddMilliseconds(float64 value)  
8 [C#]  
9 public DateTime AddMilliseconds(double value)
```

10 Summary

11 Adds a specified number of milliseconds to the value of the current
12 instance.

13 Parameters

Parameter	Description
<i>value</i>	A System.Double containing the number of whole and fractional milliseconds. For example, 4.5 is equivalent to 4 milliseconds and 5,000 ticks. <i>value</i> can be negative or positive.

17 Return Value

19 A **System.DateTime** instance set to the sum of the date and time
20 represented by the current instance and the number of milliseconds
21 represented by *value*.

22 Description

23 [Note: *value* is rounded to the nearest tick.]

24 Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	The resulting System.DateTime is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue .

1
2
3

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

3 DateTime.AddMinutes(System.Double) 4 Method

```
5 [ILASM]  
6 .method public hidebysig instance valuetype System.DateTime  
7 AddMinutes(float64 value)  
8 [C#]  
9 public DateTime AddMinutes(double value)
```

10 Summary

11 Adds a specified number of minutes to the value of the current
12 instance.

13 Parameters

Parameter	Description
<i>value</i>	A System.Double containing the number of whole and fractional minutes. For example, 4.5 is equivalent to 4 minutes, 30 seconds, 0 milliseconds, and 0 ticks. <i>value</i> can be negative or positive.

17 Return Value

19 A **System.DateTime** instance set to the sum of the date and time
20 represented by the current instance and the number of minutes
21 represented by *value*.

22 Description

23 [Note: *value* is rounded to the nearest tick.]

24 Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	The resulting System.DateTime is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue .

1
2
3

1 DateTime.AddMonths(System.Int32)

2 Method

```
3 [ILASM]  
4 .method public hidebysig instance valuetype System.DateTime  
5 AddMonths(int32 months)  
  
6 [C#]  
7 public DateTime AddMonths(int months)
```

8 Summary

9 Adds a specified number of months to the value of the current
10 instance.

11 Parameters

12
13

Parameter	Description
<i>months</i>	A System.Int32 containing the number of months. <i>months</i> can be positive or negative, and may be greater than the number of months in a year.

14
15
16

Return Value

17 A **System.DateTime** instance set to the sum of the date and time
18 represented by the current instance and *months*.

19 Description

20 This method does not change the value of the current **DateTime**
21 instance. Instead, a new **DateTime** instance is returned whose value
22 is the result of this operation.

23 Exceptions

24
25

Exception	Condition
System.ArgumentOutOfRangeException	The resulting System.DateTime is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue . -or- The <i>months</i> parameter is less than -

1
2
3

	120,000 or greater than 120,000
--	---------------------------------

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

3 DateTime.AddSeconds(System.Double) 4 Method

```
5 [ILASM]  
6 .method public hidebysig instance valuetype System.DateTime  
7 AddSeconds(float64 value)  
8 [C#]  
9 public DateTime AddSeconds(double value)
```

10 Summary

11 Adds a specified number of seconds to the value of the current
12 instance.

13 Parameters

Parameter	Description
<i>value</i>	A System.Double containing the number of whole and fractional seconds. For example, 4.5 is equivalent to 4 seconds, 500 milliseconds, and 0 ticks. <i>value</i> can be positive or negative.

16 Return Value

17 A **System.DateTime** instance set to the sum of the date and time
18 represented by the current instance and the number of seconds
19 represented by *value*.

22 Description

23 [Note: *value* is rounded to the nearest tick.]

24 Exceptions

Exception	Condition
System.ArgumentException	The resulting System.DateTime is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue .

1 DateTime.AddTicks(System.Int64) Method

```
2 [ILASM]  
3 .method public hidebysig instance valuetype System.DateTime  
4 AddTicks(int64 value)  
5 [C#]  
6 public DateTime AddTicks(long value)
```

7 Summary

8 Adds a specified number of ticks to the value of the current instance.

9 Parameters

10
11

Parameter	Description
<i>value</i>	A System.Int64 containing the number of 100-nanosecond ticks. <i>value</i> can be positive or negative.

12
13
14

Return Value

15 A **System.DateTime** instance set to the sum of the date and time
16 represented by the current instance and the time represented by
17 *value*.

18 Exceptions

19
20

Exception	Condition
System.ArgumentOutOfRangeException	The resulting System.DateTime is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue .

21
22
23

1 DateTime.AddYears(System.Int32)

2 Method

```
3 [ILASM]  
4 .method public hidebysig instance valuetype System.DateTime  
5 AddYears(int32 value)  
  
6 [C#]  
7 public DateTime AddYears(int value)
```

8 Summary

9 Adds a specified number of years to the value of the current instance.

10 Parameters

11
12

Parameter	Description
<i>value</i>	A System.Int32 containing the number of years. <i>value</i> can be positive or negative.

13
14
15

Return Value

16 A **System.DateTime** instance set to the sum of the date and time
17 represented by the current instance and the number of years
18 represented by *value*.

19 Exceptions

20
21

Exception	Condition
System.ArgumentOutOfRangeException	The resulting System.DateTime is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue .

22
23
24

1 DateTime.Compare(System.DateTime, 2 System.DateTime) Method

```
3 [ILASM]  
4 .method public hidebysig static int32 Compare(valuetype  
5 System.DateTime t1, valuetype System.DateTime t2)  
  
6 [C#]  
7 public static int Compare(DateTime t1, DateTime t2)
```

8 Summary

9 Returns the sort order of the two specified instances of
10 **System.DateTime**.

11 Parameters

12
13

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

14
15
16

Return Value

17 A **System.Int32** containing a value that reflects the sort order of the
18 two specified instances of **System.DateTime**. The following table
19 defines the conditions under which the returned value is a negative
20 number, zero, or a positive number.

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

21
22

1 DateTime.CompareTo(System.Object)

2 Method

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

8 Summary

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

11 Parameters

12
13

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

14
15
16

Return Value

17 A **System.Int32** containing a value that reflects the sort order of the
18 current instance as compared to *value*. The following table defines the
19 conditions under which the returned value is a negative number, zero,
20 or a positive number.

Value	Description
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.

21

22 Description

23 Any instance of **System.DateTime**, regardless of its value, is
24 considered greater than a null reference.

25
26
27

[Note: This method is implemented to support the **System.IComparable** interface.]

28 Exceptions

29
30

1
2
3

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

1 DateTime.DaysInMonth(System.Int32, 2 System.Int32) Method

```
3 [ILASM]  
4 .method public hidebysig static int32 DaysInMonth(int32  
5 year, int32 month)  
6  
7 [C#]  
8 public static int DaysInMonth(int year, int month)
```

8 Summary

9 Returns the number of days in a specified month of a specified year.

10 Parameters

11
12

Parameter	Description
<i>year</i>	A System.Int32 containing the year.
<i>month</i>	The month (a System.Int32 between 1 and 12).

13
14
15

14 Return Value

16 A **System.Int32** set to the number of days in the specified month for
17 the specified year. If the specified month is February, the return value
18 is 28 or 29 depending upon whether the specified year is a leap year.

19 Exceptions

20
21

Exception	Condition
System.ArgumentOutOfRangeException	<i>month</i> is less than 1 or greater than 12.

22
23
24

1 DateTime.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 Returns a **System.Boolean** indicating whether the current instance is
8 equal to a specified object.

9 Parameters

10
11

Parameter	Description
<i>value</i>	A System.Object to compare with the current instance.

12
13
14

Return Value

15 **true** if *value* is a specified **System.DateTime** instance is equal to the
16 current instance; otherwise, **false**.

17
18 [Note: This method overrides **System.Object.Equals**.]

19

1 DateTime.Equals(System.DateTime, 2 System.DateTime) Method

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

8 Summary

9 Returns a **System.Boolean** indicating whether two specified instances
10 of **System.DateTime** are equal.

11 Parameters

12
13

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

14
15
16

15 Return Value

17 **true** if the two **System.DateTime** values are equal; otherwise, **false**.

18

1 DateTime.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

10 A **System.Int32** containing the hash code for this instance.

11 Description

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

13

14 [*Note:* This method overrides **System.Object.GetHashCode.**]

15

1 DateTime.IsLeapYear(System.Int32)

2 Method

```
3 [ILASM]  
4 .method public hidebysig static bool IsLeapYear(int32 year)  
5 [C#]  
6 public static bool IsLeapYear(int year)
```

7 Summary

8 Returns a **System.Boolean** value indicating whether a specified year
9 is a leap year.

10 Parameters

11
12

Parameter	Description
<i>year</i>	A System.Int32 representing the year. <i>year</i> can be positive or negative.

13
14
15

Return Value

16 **true** if the specified year is a leap year; otherwise, **false**.

17

1 DateTime.op_Addition(System.DateTime, 2 System.TimeSpan) Method

```
3 [ILASM]  
4 .method public hidebysig static specialname valuetype  
5 System.DateTime op_Addition(valuetype System.DateTime d,  
6 valuetype System.TimeSpan t)  
  
7 [C#]  
8 public static DateTime operator +(DateTime d, TimeSpan t)
```

9 Summary

10 Adds a specified **System.TimeSpan** value to a specified
11 **System.DateTime** value.

12 Parameters

13
14

Parameter	Description
<i>d</i>	A System.DateTime value.
<i>t</i>	A System.TimeSpan value.

15
16
17

16 Return Value

18 A **System.DateTime** instance that is the sum of the values of *d* and *t*.

19 Description

20 The returned value is equivalent to **DateTime**(*d*.Ticks + *t*.Ticks).

21 Exceptions

22
23

Exception	Condition
System.ArgumentOutOfRangeException	The resulting date and time is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue .

24
25
26

1 DateTime.op_Equality(System.DateTime, 2 System.DateTime) Method

```
3 [ILASM]  
4 .method public hidebysig static specialname bool  
5 op_Equality(valuetype System.DateTime d1, valuetype  
6 System.DateTime d2)  
  
7 [C#]  
8 public static bool operator ==(DateTime d1, DateTime d2)
```

9 Summary

10 Returns a **System.Boolean** value indicating whether the two specified
11 instances of **System.DateTime** are equal.

12 Parameters

13
14

Parameter	Description
<i>d1</i>	The first System.DateTime to compare.
<i>d2</i>	The second System.DateTime to compare.

15
16
17

Return Value

18 **true** if *d1*.Ticks value is equal to the *d2*.Ticks value; otherwise, **false**.

19

1 DateTime.op_GreaterThan(System.DateTime, System.DateTime) Method

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

9 Summary

10 Returns a **System.Boolean** value indicating whether one specified
11 **System.DateTime** is greater than another specified
12 **System.DateTime**.

13 Parameters

Parameter	Description
<i>t1</i>	A System.DateTime .
<i>t2</i>	A System.DateTime .

16 Return Value

19 **true** if *t1*.Ticks value is greater than the *t2*.Ticks value; otherwise,
20 **false**.

21

1 DateTime.op_GreaterThanOrEqual(System 2 .DateTime, System.DateTime) Method

```
3 [ILASM]  
4 .method public hideby sig static specialname bool  
5 op_GreaterThanOrEqual(valuetype System.DateTime t1,  
6 valuetype System.DateTime t2)  
  
7 [C#]  
8 public static bool operator >=(DateTime t1, DateTime t2)
```

9 Summary

10 Returns a **System.Boolean** value indicating whether one specified
11 **System.DateTime** is greater than or equal to another specified
12 **System.DateTime**.

13 Parameters

14
15

Parameter	Description
<i>t1</i>	A System.DateTime .
<i>t2</i>	A System.DateTime .

16

17 Return Value

18

19 **true** if *t1*.Ticks value is greater than or equal to *t2*.Ticks value;
20 otherwise, **false**.

21

1 DateTime.op_Inequality(System.DateTime, System.DateTime) Method

```
3 [ILASM]
4 .method public hidebysig static specialname bool
5 op_Inequality(valuetype System.DateTime d1, valuetype
6 System.DateTime d2)
7
8 [C#]
9 public static bool operator !=(DateTime d1, DateTime d2)
```

9 Summary

10 Returns a **System.Boolean** value indicating whether two specified
11 instances of **System.DateTime** are not equal.

12 Parameters

13
14

Parameter	Description
<i>d1</i>	A System.DateTime .
<i>d2</i>	A System.DateTime .

15
16
17

16 Return Value

18 **true** if *d1*.Ticks value is not equal to *d2*.Ticks value; otherwise, **false**.

19

1 DateTime.op_LessThan(System.DateTime, 2 System.DateTime) Method

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

9 Summary

10 Returns a **System.Boolean** value indicating whether one specified
11 **System.DateTime** is less than another specified **System.DateTime**.

12 Parameters

13
14

Parameter	Description
<i>t1</i>	A System.DateTime .
<i>t2</i>	A System.DateTime .

15
16
17

Return Value

18 **true** if *t1*.Ticks value is less than *t2*.Ticks value; otherwise, **false**.

19

1 DateTime.op_LessThanOrEqual(System.DateTime, System.DateTime) Method

```
3 [ILASM]
4 .method public hidebysig static specialname bool
5 op_LessThanOrEqual(valuetype System.DateTime t1, valuetype
6 System.DateTime t2)
7
8 [C#]
9 public static bool operator <=(DateTime t1, DateTime t2)
```

9 Summary

10 Returns a **System.Boolean** value indicating whether one specified
11 **System.DateTime** is less than or equal to another specified
12 **System.DateTime**.

13 Parameters

Parameter	Description
<i>t1</i>	A System.DateTime .
<i>t2</i>	A System.DateTime .

16 Return Value

19 **true** if *t1*.Ticks value is less than or equal to *t2*.Ticks value; otherwise,
20 **false**.

21

1 DateTime.op_Subtraction(System.DateTime 2 me, System.TimeSpan) Method

```
3 [ILASM]  
4 .method public hidebysig static specialname valuetype  
5 System.DateTime op_Subtraction(valuetype System.DateTime d,  
6 valuetype System.TimeSpan t)  
  
7 [C#]  
8 public static DateTime operator -(DateTime d, TimeSpan t)
```

9 Summary

10 Subtracts a specified **System.TimeSpan** from a specified
11 **System.DateTime**.

12 Parameters

13
14

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

15
16
17

16 Return Value

18 A **System.DateTime** whose value is the value of *d* minus the value of
19 *t*.

20 Description

21 The returned value is equivalent to **System.DateTime**(*d*.Ticks -
22 *t*.Ticks).

23 Exceptions

24
25

Exception	Condition
System.ArgumentOutOfRangeException	The resulting date and time is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue .

26
27
28

1 DateTime.op_Subtraction(System.DateTime 2 me, System.DateTime) Method

```
3 [ILASM]  
4 .method public hidebysig static specialname valuetype  
5 System.TimeSpan op_Subtraction(valuetype System.DateTime  
6 d1, valuetype System.DateTime d2)  
  
7 [C#]  
8 public static TimeSpan operator -(DateTime d1, DateTime d2)
```

9 Summary

10 Subtracts a specified **System.DateTime** from another specified
11 **System.DateTime** value, producing a time interval.

12 Parameters

13
14

Parameter	Description
<i>d1</i>	A System.DateTime (the minuend).
<i>d2</i>	A System.DateTime (the subtrahend).

15

16 Return Value

17

18 A **System.TimeSpan** that is the time interval between *d1* and *d2*.

19 Description

20 The returned value is equivalent to **System.TimeSpan**(*d1*.Ticks -
21 *d2*.Ticks).

22 Exceptions

23

24

Exception	Condition
System.ArgumentOutOfRangeException	The resulting date and time is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue .

25

26

27

DateTime.Parse(System.String) Method

```
[ILASM]
.method public hidebysig static valuetype System.DateTime
Parse(string s)

[C#]
public static DateTime Parse(string s)
```

Summary

Returns the specified **System.String** converted to a **System.DateTime** value.

Parameters

Parameter	Description
s	A System.String containing a value to convert. The string is interpreted using the System.Globalization.DateTimeStyles.None style.

Return Value

The **System.DateTime** value obtained from s.

Description

This version of **System.DateTime.Parse** is equivalent to **System.DateTime.Parse(s, null, System.Globalization.DateTimeStyles.None)**.

The string s is parsed using the formatting information in a **System.Globalization.DateTimeFormatInfo** initialized for the current system culture.

In order for the string to be successfully parsed, it is required to represent a date and time value in one of the standard **System.DateTime** patterns described in **System.Globalization.DateTimeFormatInfo**.

If the string contains only a time, and no date, then the current date (**System.DateTime.Now**) is used. If the string contains only a date and no time, this method assumes 12 a.m.

Any leading, trailing, and inner white space characters are ignored.

1 **Exceptions**

2

3

Exception	Condition
System.ArgumentNullException	s is a null reference.
System.FormatException	s does not contain a valid string representation of a time or date and time.

4

5

6

1 DateTime.Parse(System.String, 2 System.IFormatProvider) Method

```
3 [ILASM]  
4 .method public hidebysig static valuetype System.DateTime  
5 Parse(string s, class System.IFormatProvider provider)  
  
6 [C#]  
7 public static DateTime Parse(string s, IFormatProvider  
8 provider)
```

9 Summary

10 Returns the specified **System.String** converted to a
11 **System.DateTime** value.

12 Parameters

13
14

Parameter	Description
<i>s</i>	A System.String containing the value to convert. The string is interpreted using the System.Globalization.DateTimeStyles.None style.
<i>provider</i>	A System.IFormatProvider that supplies a System.Globalization.DateTimeFormatInfo object containing culture-specific format information about <i>s</i> .

15
16

16 Return Value

17

18 The **System.DateTime** value obtained from *s*.

19 Description

20 This version of **System.DateTime.Parse** is equivalent to
21 **System.DateTime.Parse**(*s*, *provider*,
22 **System.Globalization.DateTimeStyles.None**).

23

24 The string *s* is parsed using the culture-specific formatting information
25 from the **System.Globalization.DateTimeFormatInfo** instance
26 supplied by *provider*. If *provider* is **null** or a
27 **System.Globalization.DateTimeFormatInfo** cannot be obtained
28 from *provider*, the formatting information for the current system
29 culture is used.

30

31 In order for the string to be successfully parsed, it is required to
32 represent a date and time value in one of the standard
33 **System.DateTime** patterns described in

1
2
3
4
5
6
7
8
9
10

System.Globalization.DateTimeFormatInfo.

If the string contains only a time, and no date, then the current date (**System.DateTime.Now**) is used. If the string contains only a date and no time, this method assumes 12 a.m.

Any leading, trailing, and inner white space characters are ignored.

Exceptions

Exception	Condition
System.ArgumentException	s is a null reference.
System.FormatException	s does not contain a valid string representation of a time or date and time.

11
12
13

DateTime.Parse(System.String, System.IFormatProvider, System.Globalization.DateTimeStyles) Method

```
[ILASM]
.method public hidebysig static valuetype System.DateTime
Parse(string s, class System.IFormatProvider provider,
valuetype System.Globalization.DateTimeStyles styles)

[C#]
public static DateTime Parse(string s, IFormatProvider
provider, DateTimeStyles styles)
```

Summary

Returns the specified **System.String** converted to a **System.DateTime** value.

Parameters

Parameter	Description
<i>s</i>	A System.String containing the value to convert.
<i>provider</i>	A System.IFormatProvider that supplies a System.Globalization.DateTimeFormatInfo object containing culture-specific format information about <i>s</i> .
<i>styles</i>	One or more System.Globalization.DateTimeStyles values that specify the style of <i>s</i> . Specify multiple values for <i>styles</i> using the bitwise OR operator.

Return Value

The **System.DateTime** value obtained from *s*.

Description

The string *s* is parsed using the culture-specific formatting information from the **System.Globalization.DateTimeFormatInfo** instance supplied by *provider*. If *provider* is **null** or a **System.Globalization.DateTimeFormatInfo** cannot be obtained from *provider*, the formatting information for the current system culture is used.

In order for the string to be successfully parsed, it is required to

1 represent a date and time value in one of the standard
2 **System.DateTime** patterns described in
3 **System.Globalization.DateTimeFormatInfo**.

4
5 If the string contains only a time, and no date, and if the *styles*
6 parameter is set to
7 **System.Globalization.DateTimeStyles.NoCurrentDateDefault** the
8 Gregorian year 1, month 1, day 1 are used. In all other cases where a
9 date is not specified, the current date (**System.DateTime.Now**) is
10 used.

11
12 If the string contains only a date and no time, this method assumes 12
13 a.m.

14
15 For all settings of the *styles* parameter, any leading, trailing, and inner
16 white space characters are ignored.

17 Exceptions

18

19

Exception	Condition
System.ArgumentException	s is a null reference.
System.FormatException	s does not contain a valid string representation of a time or date and time.

20

21

22

1 DateTime.ParseExact(System.String, 2 System.String, System.IFormatProvider) 3 Method

```
4 [ILASM]  
5 .method public hidebysig static valuetype System.DateTime  
6 ParseExact(string s, string format, class  
7 System.IFormatProvider provider)
```

```
8 [C#]  
9 public static DateTime ParseExact(string s, string format,  
10 IFormatProvider provider)
```

11 Summary

12 Converts the specified **System.String** representation of a date and
13 time to its **System.DateTime** equivalent using a specified format and
14 **System.IFormatProvider**.

15 Parameters

16
17

Parameter	Description
<i>s</i>	A System.String containing a date and time to convert. The format of the string is required to match the specified format exactly.
<i>format</i>	A System.String containing the expected format of <i>s</i> . [Note: For a list of valid <i>format</i> values, see System.Globalization.DateTimeFormatInfo .]
<i>provider</i>	A System.IFormatProvider that supplies a System.Globalization.DateTimeFormatInfo object containing culture-specific format information about <i>s</i> .

18
19
20

Return Value

21 A **System.DateTime** equivalent to the date and time contained in *s*.

22 Description

23 **System.DateTime.ParseExact** constructs a **System.DateTime** from
24 the string *s*. The string is required to specify a date and, optionally, a
25 time in the specified format.

26
27
28
29
30

The string *s* is parsed using the culture-specific formatting information from the **System.Globalization.DateTimeFormatInfo** instance supplied by *provider*. If *provider* is **null** or a **System.Globalization.DateTimeFormatInfo** cannot be obtained

1 from *provider*, the formatting information for the current system
 2 culture is used.
 3
 4 If the *s* string contains only a time, and no date, then the current date
 5 (**System.DateTime.Now**) is used. If the string contains only a date
 6 and no time, this method assumes 12 a.m.
 7
 8 Leading, trailing, and inner white space characters are not allowed.
 9
 10 [Note: For information on formatting system-supplied data types, see
 11 the **System.IFormattable** interface.]

12 **Exceptions**
 13
 14

Exception	Condition
System.ArgumentNullException	<i>s</i> or <i>format</i> is a null reference.
System.FormatException	<i>s</i> or <i>format</i> is an empty string.
	-or- <i>s</i> does not contain a date and time that were recognized as the pattern specified in <i>format</i> .

15
 16 **Example**
 17

18 This example demonstrates the **System.DateTime.ParseExact**
 19 method.

```
20 [C#]
21
22 using System;
23 using System.Globalization;
24
25 public class DateTimeTest {
26     public static void Main() {
27         DateTimeFormatInfo dtfi = new DateTimeFormatInfo();
28
29         DateTime dt = DateTime.ParseExact("January 22",
30 dtfi.MonthDayPattern, null);
31         Console.WriteLine(dt);
32     }
33 }
34
```

1
2
3

The output is

1/22/2001 12:00:00 AM

4

1 DateTime.ParseExact(System.String, 2 System.String, System.IFormatProvider, 3 System.Globalization.DateTimeStyles) 4 Method

```
5 [ILASM]  
6 .method public hidebysig static valuetype System.DateTime  
7 ParseExact(string s, string format, class  
8 System.IFormatProvider provider, valuetype  
9 System.Globalization.DateTimeStyles style)
```

```
10 [C#]  
11 public static DateTime ParseExact(string s, string format,  
12 IFormatProvider provider, DateTimeStyles style)
```

13 Summary

14 Converts the **System.String** representation of a date and time to its
15 **System.DateTime** equivalent using a specified style, the expected
16 format, and culture-specific format information.

17 Parameters

Parameter	Description
<i>s</i>	A System.String containing a date and time to convert. The format of the string is required to match the specified format exactly.
<i>format</i>	A System.String containing the expected format of <i>s</i> . [Note: For a list of valid <i>format</i> values, see System.Globalization.DateTimeFormatInfo .]
<i>provider</i>	A System.IFormatProvider that supplies a System.Globalization.DateTimeFormatInfo object containing culture-specific format information about <i>s</i> .
<i>style</i>	One or more System.Globalization.DateTimeStyles values that specify the style of <i>s</i> . Specify multiple values for <i>styles</i> using the bitwise OR operator.

21 Return Value

23 A **System.DateTime** equivalent to the date and time contained in *s*.

24 Description

25 **System.DateTime.ParseExact** constructs a **System.DateTime** from
26 the string *s*. The string is required to specify a date and, optionally, a

1 time in the provided format.

2
3 The string *s* is parsed using the culture-specific formatting information
4 from the **System.Globalization.DateTimeFormatInfo** instance
5 supplied by *provider*. If *provider* is **null** or a
6 **System.Globalization.DateTimeFormatInfo** cannot be obtained
7 from *provider*, the formatting information for the current system
8 culture is used.

9
10 If the *s* string contains only a time, and no date, and if the *styles*
11 parameter is set to
12 **System.Globalization.DateTimeStyles.NoCurrentDateDefault** the
13 Gregorian year 1, month 1, day 1 are used, and no leading, trailing, or
14 inner white space characters are allowed. In all other cases where a
15 date is not specified, the current date (**System.DateTime.Now**) is
16 used.

17
18 If the *s* string contains only a date and no time, this method assumes
19 12 a.m.

20
21 [Note: For information on formatting system-supplied data types, see
22 the **System.IFormattable** interface.]

23 Exceptions

Exception	Condition
System.ArgumentNullException	<i>s</i> or <i>format</i> is a null reference.
System.FormatException	<i>s</i> or <i>format</i> is an empty string.
	-or- <i>s</i> does not contain a date and time that were recognized as the pattern specified in <i>format</i> .

27 Example

29 This example demonstrates the **System.DateTime.ParseExact**
30 method.

31 [C#]
32

```
33 using System;  
34 using System.Globalization;  
35  
36 public class DateTimeTest {  
37     public static void Main() {  
38         DateTimeFormatInfo dtfi = new DateTimeFormatInfo();  
39     }  
}
```

```
1         DateTime dt = DateTime.ParseExact(" January 22 ",
2 dtfi.MonthDayPattern, null,
3 DateTimeStyles.AllowWhiteSpaces);
4         Console.WriteLine(dt);
5     }
6 }
7
```

```
8     The output is
9
10    1/22/2001 12:00:00 AM
11
```

12

1 **DateTime.ParseExact(System.String,**
2 **System.String[], System.IFormatProvider,**
3 **System.Globalization.DateTimeStyles)**
4 **Method**

```
5 [ILASM]  
6 .method public hidebysig static valuetype System.DateTime  
7 ParseExact(string s, class System.String[] formats, class  
8 System.IFormatProvider provider, valuetype  
9 System.Globalization.DateTimeStyles style)
```

```
10 [C#]  
11 public static DateTime ParseExact(string s, string[]  
12 formats, IFormatProvider provider, DateTimeStyles style)
```

13 **Summary**

14 Converts the **System.String** representation of a date and time to its
15 **System.DateTime** equivalent using a specified style, an array of
16 expected formats, and culture-specific format information.

17 **Parameters**

Parameter	Description
<i>s</i>	A System.String containing one or more dates and times to convert. The format of the string is required to match the specified format exactly.
<i>formats</i>	A System.String array containing the expected formats of <i>s</i> . [Note: For a list of valid <i>format</i> values, see System.Globalization.DateTimeFormatInfo .]
<i>provider</i>	A System.IFormatProvider that supplies a System.Globalization.DateTimeFormatInfo object containing culture-specific format information about <i>s</i> .
<i>style</i>	One or more System.Globalization.DateTimeStyles values that specify the style of <i>s</i> . Specify multiple values for <i>styles</i> using the bitwise OR operator.

20
21 **Return Value**
22

23 A **System.DateTime** equivalent to the date and time contained in *s*.

24 **Description**

1 **System.DateTime.ParseExact** constructs a **System.DateTime** from
2 the **s** **System.String**. The string is required to specify a date and,
3 optionally, a time in the provided format.

4
5 The string *s* is parsed using the culture-specific formatting information
6 from the **System.Globalization.DateTimeFormatInfo** instance
7 supplied by *provider*. If *provider* is **null** or a
8 **System.Globalization.DateTimeFormatInfo** cannot be obtained
9 from *provider*, the formatting information for the current system
10 culture is used.

11
12 If the *s* string contains only a time, and no date, and if the *styles*
13 parameter is set to
14 **System.Globalization.DateTimeStyles.NoCurrentDateDefault** the
15 Gregorian year 1, month 1, day 1 are used, and no leading, trailing, or
16 inner white space characters are allowed. In all other cases where a
17 date is not specified, the current date (**System.DateTime.Now**) is
18 used.

19
20 If the *s* string contains only a date and no time, this method assumes
21 12 a.m.

22
23 [Note: For information on formatting system-supplied data types, see
24 the **System.IFormattable** interface.]

25 Exceptions

26
27

Exception	Condition
System.ArgumentNullException	<i>s</i> or <i>formats</i> is a null reference.
System.FormatException	<i>s</i> or <i>format</i> is an empty string.
	-or- <i>s</i> does not contain a date and time that were recognized as the pattern specified in <i>format</i>

28
29 **Example**
30

31 This example demonstrates the **System.DateTime.ParseExact**
32 method.

33
34

[C#]

```
35 using System;  
36 using System.Globalization;  
37  
38 public class DateTimeTest {  
39     public static void Main() {
```

```
1         DateTimeFormatInfo dtfi = new DateTimeFormatInfo();
2         string [] patterns = {dtfi.LongTimePattern,
3 dtfi.ShortTimePattern};
4
5         DateTime dt = DateTime.ParseExact("10:11:12",
6 patterns, null, DateTimeStyles.NoCurrentDateDefault);
7         Console.WriteLine(dt);
8     }
9 }
10
```

```
11     The output is
12
13     1/1/0001 10:11:12 AM
```

```
14
```

1 DateTime.Subtract(System.DateTime)

2 Method

```
3 [ILASM]  
4 .method public hidebysig instance valuetype System.TimeSpan  
5 Subtract(valuetype System.DateTime value)  
6  
7 [C#]  
8 public TimeSpan Subtract(DateTime value)
```

8 Summary

9 Subtracts a specified date and time from the current instance.

10 Parameters

11
12

Parameter	Description
<i>value</i>	An instance of System.DateTime .

13
14
15

14 Return Value

16 A **System.TimeSpan** interval equal to the date and time represented
17 by the current instance minus the date and time represented by the
18 specified **System.DateTime**.

19

1 DateTime.Subtract(System.TimeSpan) 2 Method

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

8 Summary

9 Subtracts a specified **System.TimeSpan** from the current instance.

10 Parameters

11
12

Parameter	Description
<i>value</i>	An instance of System.TimeSpan .

13
14
15

Return Value

16 A new **System.DateTime** instance equal to the date and time
17 represented by the current instance minus the time interval of the
18 specified **System.TimeSpan**.

19 Exceptions

20
21

Exception	Condition
System.ArgumentOutOfRangeException	The resulting date and time is less than System.DateTime.MinValue or greater than System.DateTime.MaxValue .

22
23
24

1 DateTime.ToLocalTime() Method

```
2 [ILASM]  
3 .method public hidebysig instance valuetype System.DateTime  
4 ToLocalTime()  
5 [C#]  
6 public DateTime ToLocalTime()
```

7 Summary

8 Converts the universal time coordinate (UTC) time value in the current
9 instance to local time.

10 Return Value

11

12 An instance of **System.DateTime** equivalent of the time value in the
13 current instance, adjusted to the local time zone and daylight saving
14 time. If the result is too large or too small to be represented as a
15 **System.DateTime**, this method returns a **System.DateTime** set to
16 **System.DateTime.MaxValue** or **System.DateTime.MinValue**.

17 Description

18 This method assumes that the current instance of **System.DateTime**
19 holds the UTC time value, and not a local time. Each time it is invoked,
20 this method performs the necessary modifications on the
21 **System.DateTime** to derive the local time, whether the current
22 **System.DateTime** holds the UTC time or not.

23

24 The local time zone information is obtained from the operating system.

25

1 DateTime.ToLongDateString() Method

```
2 [ILASM]  
3 .method public hidebysig instance string ToLongDateString()  
4 [C#]  
5 public string ToLongDateString()
```

6 Summary

7 Converts the date denoted by the current instance to its equivalent
8 long date **System.String** representation.

9 Return Value 10

11 A **System.String** containing the same value as a **System.String**
12 returned by **System.DateTime.ToString** ("D", null).

13 Description

14 The value of the current instance is formatted using the long date
15 format specifier, 'D'.
16

17 [*Note:* This format uses the culture of the current thread. To specify
18 formatting using a different culture, use **System.DateTime.ToString**.
19

20 For more information regarding the long date specifier, see
21 **System.Globalization.DateTimeFormatInfo**.]
22

1 DateTime.ToLongTimeString() Method

```
2 [ILASM]
3 .method public hidebysig instance string ToLongTimeString()
4 [C#]
5 public string ToLongTimeString()
```

6 Summary

7 Converts the time denoted by the current instance to its equivalent
8 long time **System.String** representation.

9 Return Value

10

11 A **System.String** containing the same value as a **System.String**
12 returned by **System.DateTime.ToString** ("T", null).

13 Description

14 The value of the current instance is formatted using the long time
15 format specifier, 'T'.

16

17 [*Note:* This format uses the culture of the current thread. To specify
18 formatting using a different culture, use **System.DateTime.ToString**.

19

20 For more information regarding the long time specifier, see
21 **System.Globalization.DateTimeFormatInfo**.]

22

1 DateTime.ToShortDateString() Method

```
2 [ILASM]  
3 .method public hidebysig instance string  
4 ToShortDateString()  
5 [C#]  
6 public string ToShortDateString()
```

7 Summary

8 Converts the date denoted by the current instance to its equivalent
9 short date **System.String** representation.

10 Return Value

11

12 A **System.String** containing the same value as a **System.String**
13 returned by **System.DateTime.ToString** ("d", null).

14 Description

15 The value of the current instance is formatted using the long time
16 format specifier, 'd'.
17

18 [*Note:* This format uses the culture of the current thread. To specify
19 formatting using a different culture, use **System.DateTime.ToString**.
20

21 For more information regarding the long time specifier, see
22 **System.Globalization.DateTimeFormatInfo**.]
23

1 DateTime.ToShortTimeString() Method

```
2 [ILASM]  
3 .method public hidebysig instance string  
4 ToShortTimeString()  
5 [C#]  
6 public string ToShortTimeString()
```

7 Summary

8 Converts the time denoted by this instance to its equivalent short time
9 **System.String** representation.

10 Return Value

11

12 A **System.String** containing the same value as a **System.String**
13 returned by **System.DateTime.ToString** ("t", null).

14 Description

15 The value of the current instance is formatted using the long time
16 format specifier, 't'.
17

18 [*Note:* This format uses the culture of the current thread. To specify
19 formatting using a different culture, use **System.DateTime.ToString**.
20

21 For more information regarding the long time specifier, see
22 **System.Globalization.DateTimeFormatInfo**.]
23

1 DateTime.ToString(System.IFormatProvid 2 er) Method

```
3 [ILASM]  
4 .method public final hidebysig virtual string  
5 ToString(class System.IFormatProvider provider)  
  
6 [C#]  
7 public string ToString(IFormatProvider provider)
```

8 Summary

9 Returns a **System.String** representation of the value of the current
10 instance.

11 Parameters

12
13

Parameter	Description
<i>provider</i>	A System.IFormatProvider that supplies a System.Globalization.DateTimeFormatInfo containing culture-specific formatting information.

14
15
16

Return Value

17 A **System.String** representation of the current instance formatted
18 using the general format specifier, ("G"). The string takes into account
19 the formatting information in the
20 **System.Globalization.DateTimeFormatInfo** instance supplied by
21 *provider*.

22 Description

23 This version of **System.DateTime.ToString** is equivalent to
24 **System.DateTime.ToString** ("G", *provider*).

25
26 If *provider* is **null** or the
27 **System.Globalization.DateTimeFormatInfo** cannot be obtained
28 from *provider*, the formatting information for the current system
29 culture is used.

30
31 [Note: The general format specifier ("G") provides the general date
32 pattern including the long time form, equivalent to
33 **System.Globalization.DateTimeFormatInfo.ShortDatePattern**
34 combined with
35 **System.Globalization.DateTimeFormatInfo.LongTimePattern**.
36 For more information on format specifiers, see
37 **System.Globalization.DateTimeFormatInfo**. For information on

1 formatting system-supplied data types, see the
2 **System.IFormattable** interface.]

3

1 DateTime.ToString(System.String, 2 System.IFormatProvider) Method

```
3 [ILASM]  
4 .method public final hidebysig virtual string  
5 ToString(string format, class System.IFormatProvider  
6 provider)  
  
7 [C#]  
8 public string ToString(string format, IFormatProvider  
9 provider)
```

10 Summary

11 Returns a **System.String** representation of the value of the current
12 instance.

13 Parameters

14
15

Parameter	Description
<i>format</i>	A System.String containing a character that specifies the format of the returned string. [Note: For a list of valid values, see System.Globalization.DateTimeFormatInfo .]
<i>provider</i>	A System.IFormatProvider that supplies a System.Globalization.DateTimeFormatInfo instance containing culture-specific formatting information.

16
17
18

Return Value

19 A **System.String** representation of the current instance formatted as
20 specified by *format*. The string takes into account the information in
21 the **System.Globalization.DateTimeFormatInfo** supplied by
22 *provider*.

23 Description

24 If *provider* is **null** or a **System.Globalization.DateTimeFormatInfo**
25 cannot be obtained from *provider*, the formatting information for the
26 current system culture is used.

27
28
29

If *format* is a null reference, the general format specifier "G" is used.

30 [Note: For more information regarding the standard format specifier,
31 see **System.Globalization.DateTimeFormatInfo**. For information
32 on formatting system-supplied data types, see the
33 **System.IFormattable** interface.
34

1 This method is implemented to support the **System.IFormattable**
2 interface.]

3 **Exceptions**

4
5

Exception	Condition
System.FormatException	The length of the <i>format</i> string is 1, and it is not one of the format specifier characters defined for System.Globalization.DateTimeFormatInfo . -or- The <i>format</i> string does not contain a valid custom format pattern.

6
7
8

1 DateTime.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
8 instance.

9 Return Value

10

11 A **System.String** representation of the current instance formatted
12 using the general format specifier, ("G"). The string takes into account
13 the current system culture.

14 Description

15 This version of **System.DateTime.ToString** is equivalent to
16 **System.DateTime.ToString** ("G", null).

17

18 [*Note:* For more information about the general format specifier ("G")
19 see **System.Globalization.DateTimeFormatInfo**.

20

21 This method overrides **System.Object.ToString**.]

22

1 DateTime.ToString(System.String)

2 Method

3 [ILASM]
4 .method public hidebysig instance string ToString(string
5 format)

6 [C#]
7 public string ToString(string format)

8 Summary

9 Returns a **System.String** representation of the value of the current
10 instance.

11 Parameters

12
13

Parameter	Description
<i>format</i>	A System.String that specifies the format of the returned string. [Note: For a list of valid values, see System.Globalization.DateTimeFormatInfo .]

14
15
16

Return Value

17 A **System.String** representation of the current instance formatted as
18 specified by *format*. The string takes into account the current system
19 culture.

20 Description

21 This version of **System.DateTime.ToString** is equivalent to
22 **System.DateTime.ToString** (*format*, **null**).

23
24 If *format* is a null reference, the general format specifier "G" is used.

25
26 [Note: This method uses the culture information of the current thread.

27
28 For information on formatting system-supplied data types, see the
29 **System.IFormattable** interface.]

30 Exceptions

31
32

Exception	Condition
System.FormatException	The length of the <i>format</i> string is 1, and it is not one of the format specifier characters defined for

1
2
3

System.Globalization.DateTimeFormatInfo.

-or-

The *format* string does not contain a valid custom format pattern.

1 DateTime.ToUniversalTime() Method

```
2 [ILASM]  
3 .method public hidebysig instance valuetype System.DateTime  
4 ToUniversalTime()  
5 [C#]  
6 public DateTime ToUniversalTime()
```

7 Summary

8 Converts the current **System.DateTime** value to coordinated
9 universal time (UTC).

10 Return Value

11

12 The UTC **System.DateTime** equivalent of the current
13 **System.DateTime** value. If the result is too large or too small to be
14 represented as a **System.DateTime**, the current function returns a
15 **System.DateTime** set to **System.DateTime.MaxValue** or
16 **System.DateTime.MinValue**.

17 Description

18 This method assumes that the current instance of **System.DateTime**
19 holds the local time value, and not a UTC time. Therefore each time it
20 is run, this method performs the necessary modifications on the
21 **System.DateTime** to derive the UTC time, whether the current
22 **System.DateTime** holds the local time or not.

23

24 The local time zone information is obtained from the operating system.

25

1 DateTime.Date Property

```
2 [ILASM]
3 .property valuetype System.DateTime Date { public hidebysig
4 specialname instance valuetype System.DateTime get_Date() }
5 [C#]
6 public DateTime Date { get; }
```

7 Summary

8 Gets the date component of the current instance.

9 Property Value

10

11 A new **System.DateTime** instance with the same date as the current
12 instance, and the time value set to midnight (00:00:00).

13 Description

14 This property is read-only.

15

1 DateTime.Day Property

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

7 Summary

8 Gets the day of the month represented by the current instance.

9 Property Value

10

11 A **System.Int32** between 1 and 31 set to the day of the month
12 component of the current instance.

13 Description

14 This property is read-only.

15

1 DateTime.DayOfYear Property

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

7 Summary

8 Gets the day of the year represented by the current instance.

9 Property Value

10

11 A **System.Int32** between 1 and 366 set to the day of the year
12 component of the current instance.

13 Description

14 This property is read-only.

15

1 DateTime.Hour Property

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

7 Summary

8 Gets the hour represented by the current instance.

9 Property Value

10

11 A **System.Int32** between 0 and 23 set to the hour component of the
12 current instance.

13 Description

14 This property is read-only.

15

1 DateTime.Millisecond Property

```
2 [ILASM]
3 .property int32 Millisecond { public hidebysig specialname
4 instance int32 get_Millisecond() }
5
6 [C#]
7 public int Millisecond { get; }
```

7 Summary

8 Gets the milliseconds component of the date represented by the
9 current instance.

10 Property Value

11

12 A **System.Int32** between 0 and 999 set to the milliseconds
13 component of the current instance.

14 Description

15 This property is read-only.

16

1 DateTime.Minute Property

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

7 Summary

8 Gets the minute component of the date represented by the current
9 instance.

10 Property Value

11

12 A **System.Int32** between 0 and 59 set to the minute component of
13 the current instance.

14 Description

15 This property is read-only.

16

1 DateTime.Month Property

```
2 [ILASM]
3 .property int32 Month { public hidebysig specialname
4 instance int32 get_Month() }
5
6 [C#]
7 public int Month { get; }
```

7 Summary

8 Gets the month component of the date represented by the current
9 instance.

10 Property Value

11

12 A **System.Int32** between 1 and 12 set to the month component of
13 the current instance.

14 Description

15 This property is read-only.

16

1 DateTime.Now Property

```
2 [ILASM]  
3 .property valuetype System.DateTime Now { public hideby sig  
4 static specialname valuetype System.DateTime get_Now() }  
5 [C#]  
6 public static DateTime Now { get; }
```

7 Summary

8 Gets a **System.DateTime** representing the current local date and
9 time.

10 Description

11 The resolution of this property depends on the system timer.

12
13 This property is read-only.

14

1 DateTime.Second Property

```
2 [ILASM]
3 .property int32 Second { public hidebysig specialname
4 instance int32 get_Second() }
5
6 [C#]
7 public int Second { get; }
```

7 Summary

8 Gets the seconds component of the date represented by the current
9 instance.

10 Property Value

11

12 A **System.Int32** between 0 and 59 set to the seconds component of
13 the current instance.

14 Description

15 This property is read-only.

16

1 DateTime.Ticks Property

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

7 Summary

8 Gets the number of 100-nanosecond ticks that represent the date and
9 time of the current instance.

10 Property Value

11

12 A **System.Int64** set to the number of ticks that represent the date
13 and time of the current instance.

14 Description

15 The value of this property is the number of 100-nanosecond intervals
16 that have elapsed since 00:00:00, 1/1/0001. The value of the property
17 is between **System.DateTime.MinValue** and
18 **System.DateTime.MaxValue**.

19 This property is read-only.
20

21

1 DateTime.TimeOfDay Property

```
2 [ILASM]
3 .property valuetype System.TimeSpan TimeOfDay { public
4 hidebysig specialname instance valuetype System.TimeSpan
5 get_TimeOfDay() }
6
7 [C#]
8 public TimeSpan TimeOfDay { get; }
```

8 Summary

9 Gets the time of day of the current instance.

10 Property Value

11

12 A **System.TimeSpan** instance set to the time component of the
13 current instance.

14 Description

15 This property is read-only.

16

1 DateTime.Today Property

```
2 [ILASM]
3 .property valuetype System.DateTime Today { public
4 hidebysig static specialname valuetype System.DateTime
5 get_Today() }
6
7 [C#]
8 public static DateTime Today { get; }
```

8 Summary

9 Gets the current date.

10 Property Value

11

12 A **System.DateTime** instance set to the date of the current instance,
13 with the time set to 00:00:00.

14 Description

15 This property is read-only.

16

1 DateTime.UtcNow Property

```
2 [ILASM]
3 .property valuetype System.DateTime.UtcNow { public
4 hidebysig static specialname valuetype System.DateTime
5 get_UTCNow() }
6
7 [C#]
8 public static DateTime.UtcNow { get; }
```

8 Summary

9 Gets the current time converted to coordinated universal time (UTC).

10 Property Value

11

12 A **System.DateTime** instance set to the current date and time in
13 coordinated universal time (UTC).

14 Description

15 This property is read-only.

16

1 DateTime.Year Property

```
2 [ILASM]
3 .property int32 Year { public hidebysig specialname
4 instance int32 get_Year() }
5
6 [C#]
7 public int Year { get; }
```

7 Summary

8 Gets the year component of the date represented by the current
9 instance.

10 Property Value

11

12 A **System.Int32** between 1 and 9999 set to the year component of
13 the current instance.

14 Description

15 This property is read-only.

16