System.DateTime Structure

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

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

Assembly Info:

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

Implements:

- **System.IComparable**
- **System.IFormattable**
- **System.IComparable<System.DateTime>**
- **System.IEquatable<System.DateTime>**

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 31241376000000000L 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 expressed in units such as ticks, seconds, or instances of `System.TimeSpan`. Methods and properties in this value type take into account details such as leap years and the number of days in a month.

12:00:00 AM is Midnight.

]

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

```
[ILAsm]
public rtspecialname specialname instance void .ctor(int32 year, int32
month, int32 day, int32 hour, int32 minute, int32 second, int32
millisecond)

[C#]
public DateTime(int year, int month, int day, int hour, int minute, int
second, int millisecond)
```

Summary

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

Parameters

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

Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	*year* is less than 1 or greater than 9999 -or- *month* is less than 1 or greater than 12

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

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

```
[ILAsm]
public rtspecialname specialname instance void .ctor(int32 year, int32
month, int32 day, int32 hour, int32 minute, int32 second)

[C#]
public DateTime(int year, int month, int day, int hour, int minute, int
second)
```

Summary

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

Parameters

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

Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	*year* is less than 1 or greater than 9999 -or- *month* is less than 1 or greater than 12 -or- *day* is less than 1 or greater than the number

	of days in *month* -or- *hour* is less than 0 or greater than 23 -or- *minute* is less than 0 or greater than 59 -or- *second* is less than 0 or greater than 59

DateTime(System.Int64) Constructor

```
[ILAsm]
public rtspecialname specialname instance void .ctor(int64 ticks)

[C#]
public DateTime(long ticks)
```

Summary

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

Parameters

Parameter	Description
ticks	A `System.Int64`containing the date and time expressed in 100-nanosecond units.

Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	The date and time represented by *ticks* is less than `System.DateTime.MinValue` or greater than `System.DateTime.MaxValue`.

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

```
[ILAsm]
public rtspecialname specialname instance void .ctor(int32 year, int32
month, int32 day)

[C#]
public DateTime(int year, int month, int day)
```

Summary

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

Parameters

Parameter	Description
year	A `System.Int32` containing the year (1 through 9999).
month	A `System.Int32` containing the month (1 through 12).
day	A `System.Int32` containing the day (1 through the number of days in *month*).

Description

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

Exceptions

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

1

DateTime.MaxValue Field

[ILAsm]
```
.field public static initOnly valuetype System.DateTime MaxValue
```

[C#]
```
public static readonly DateTime MaxValue
```

Summary

A constant representing the largest possible value of `System.DateTime`.

Description

This field is read-only.

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

DateTime.MinValue Field

[ILAsm]
```
.field public static initOnly valuetype System.DateTime MinValue
```

[C#]
```
public static readonly DateTime MinValue
```

Summary

A constant representing the smallest possible value of `System.DateTime`.

Description

This field is read-only.

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

DateTime.Add(System.TimeSpan) Method

```
[ILAsm]
.method public hidebysig instance valuetype System.DateTime Add(valuetype
System.TimeSpan value)

[C#]
public DateTime Add(TimeSpan value)
```

Summary

Adds the value of a specified `System.TimeSpan` instance to the current instance.

Parameters

Parameter	**Description**
value	A `System.TimeSpan` instance.

Return Value

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

Description

A specified `System.TimeSpan` is added to the current instance of `System.DateTime`, and the result is returned as a new `System.DateTime`.

Exceptions

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

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

DateTime.AddDays(System.Double) Method

```
[ILAsm]
.method public hidebysig instance valuetype System.DateTime
AddDays(float64 value)

[C#]
public DateTime AddDays(double value)
```

Summary

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

Parameters

Parameter	Description
value	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. *value* can be negative or positive.

Return Value

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

Description

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

Exceptions

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

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

DateTime.AddHours(System.Double) Method

```
[ILAsm]
.method public hidebysig instance valuetype System.DateTime
AddHours(float64 value)

[C#]
public DateTime AddHours(double value)
```

Summary

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

Parameters

Parameter	Description
value	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. *value* can be negative or positive.

Return Value

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

Description

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

Exceptions

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

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

DateTime.AddMilliseconds(System.Double) Method

```
[ILAsm]
.method public hidebysig instance valuetype System.DateTime
AddMilliseconds(float64 value)

[C#]
public DateTime AddMilliseconds(double value)
```

Summary

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

Parameters

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

Return Value

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

Description

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

Exceptions

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

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

DateTime.AddMinutes(System.Double) Method

```
[ILAsm]
.method public hidebysig instance valuetype System.DateTime
AddMinutes(float64 value)

[C#]
public DateTime AddMinutes(double value)
```

Summary

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

Parameters

Parameter	Description
value	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. *value* can be negative or positive.

Return Value

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

Description

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

Exceptions

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

DateTime.AddMonths(System.Int32) Method

```
[ILAsm]
.method public hidebysig instance valuetype System.DateTime
AddMonths(int32 months)

[C#]
public DateTime AddMonths(int months)
```

Summary

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

Parameters

Parameter	Description
months	A `System.Int32` containing the number of months. *months* can be positive or negative, and can be greater than the number of months in a year.

Return Value

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

Description

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

Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	The resulting `System.DateTime` is less than `System.DateTime.MinValue` or greater than `System.DateTime.MaxValue`. -or- The *months* parameter is less than -120,000 or greater than 120,000

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

DateTime.AddSeconds(System.Double) Method

```
[ILAsm]
.method public hidebysig instance valuetype System.DateTime
AddSeconds(float64 value)

[C#]
public DateTime AddSeconds(double value)
```

Summary

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

Parameters

Parameter	Description
value	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. *value* can be positive or negative.

Return Value

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

Description

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

Exceptions

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

DateTime.AddTicks(System.Int64) Method

```
[ILAsm]
.method public hidebysig instance valuetype System.DateTime AddTicks(int64
value)

[C#]
public DateTime AddTicks(long value)
```

Summary

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

Parameters

Parameter	Description
value	A `System.Int64` containing the number of 100-nanosecond ticks. *value* can be positive or negative.

Return Value

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

Exceptions

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

DateTime.AddYears(System.Int32) Method

```
[ILAsm]
.method public hidebysig instance valuetype System.DateTime AddYears(int32
value)

[C#]
public DateTime AddYears(int value)
```

Summary

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

Parameters

Parameter	Description
value	A `System.Int32` containing the number of years. *value* can be positive or negative.

Return Value

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

Exceptions

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

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

```
[ILAsm]
.method public hidebysig static int32 Compare(valuetype System.DateTime
t1, valuetype System.DateTime t2)

[C#]
public static int Compare(DateTime t1, DateTime t2)
```

Summary

Returns the sort order of the two specified instances of `System.DateTime`.

Parameters

Parameter	Description
t1	The first `System.DateTime`.
t2	The second `System.DateTime`.

Return Value

The return value is a negative number, zero, or a positive number reflecting the sort order of the two specified instances of `System.DateTime`. For non-zero return values, the exact value returned by this method is unspecified. The following table defines the return value:

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

DateTime.CompareTo(System.DateTime) Method

```
[ILAsm]
.method public final hidebysig virtual int32 CompareTo(valuetype System.DateTime value)

[C#]
public int CompareTo(DateTime value)
```

Summary

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

Parameters

Parameter	Description
value	The `System.DateTime` to compare to the current instance.

Return Value

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

Value	Description
A negative number	Current instance < *value*.
Zero	Current instance == *value*.
A positive number	Current instance > *value*.

Description

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

DateTime.CompareTo(System.Object) Method

```
[ILAsm]
.method public final hidebysig virtual int32 CompareTo(object value)

[C#]
public int CompareTo(object value)
```

Summary

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

Parameters

Parameter	Description
value	The `System.Object` to compare to the current instance.

Return Value

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

Value	Description
A negative number	Current instance < *value*.
Zero	Current instance == *value*.
A positive number	Current instance > *value*, or value is a null reference.

Description

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

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

Exceptions

Exception	Condition
System.ArgumentException	*value* is not a `System.DateTime` and is not a null reference.

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

```
[ILAsm]
.method public hidebysig static int32 DaysInMonth(int32 year, int32 month)

[C#]
public static int DaysInMonth(int year, int month)
```

Summary

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

Parameters

Parameter	**Description**
year	A `System.Int32` containing the year.
month	The month (a `System.Int32` between 1 and 12).

Return Value

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

Exceptions

Exception	**Condition**
System.ArgumentOutOfRangeException	*month* is less than 1 or greater than 12.

DateTime.Equals(System.DateTime) Method

```
[ILAsm]
.method public hidebysig virtual bool Equals(valuetype System.DateTime
value)

[C#]
public override bool Equals(DateTime value)
```

Summary

Returns a `System.Boolean` indicating whether the current instance is equal to the specified DateTime.

Parameters

Parameter	Description
value	A `System.DateTime` to compare with the current instance.

Return Value

`true` if *value* is equal to the current instance; otherwise, `false`.

Description

[*Note:* This method is implemented to support the `System.IEquatable<System.DateTime>` interface.]

DateTime.Equals(System.Object) Method

[ILAsm]
```
.method public hidebysig virtual bool Equals(object value)
```

[C#]
```
public override bool Equals(object value)
```

Summary

Returns a `System.Boolean` indicating whether the current instance is equal to a specified object.

Parameters

Parameter	Description
value	A `System.Object` to compare with the current instance.

Return Value

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

Description

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

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

```
[ILAsm]
.method public hidebysig static bool Equals(valuetype System.DateTime t1,
valuetype System.DateTime t2)

[C#]
public static bool Equals(DateTime t1, DateTime t2)
```

Summary

Returns a `System.Boolean` indicating whether two specified instances of `System.DateTime` are equal.

Parameters

Parameter	Description
t1	The first `System.DateTime`.
t2	The second `System.DateTime`.

Return Value

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

DateTime.GetHashCode() Method

[ILAsm]
```
.method public hidebysig virtual int32 GetHashCode()
```

[C#]
```
public override int GetHashCode()
```

Summary

Generates a hash code for the current instance.

Return Value

A `System.Int32` containing the hash code for this instance.

Description

The algorithm used to generate the hash code is unspecified.

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

DateTime.IsLeapYear(System.Int32) Method

```
[ILAsm]
.method public hidebysig static bool IsLeapYear(int32 year)

[C#]
public static bool IsLeapYear(int year)
```

Summary

Returns a `System.Boolean` value indicating whether a specified year is a leap year.

Parameters

Parameter	Description
year	A `System.Int32` representing the year. *year* can be positive or negative.

Return Value

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

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

```
[ILAsm]
.method public hidebysig static specialname valuetype System.DateTime
op_Addition(valuetype System.DateTime d, valuetype System.TimeSpan t)

[C#]
public static DateTime operator +(DateTime d, TimeSpan t)
```

Summary

Adds a specified `System.TimeSpan` value to a specified `System.DateTime` value.

Parameters

Parameter	Description
d	A `System.DateTime` value.
t	A `System.TimeSpan` value.

Return Value

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

Description

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

Exceptions

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

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

```
[ILAsm]
.method public hidebysig static specialname bool op_Equality(valuetype
System.DateTime d1, valuetype System.DateTime d2)

[C#]
public static bool operator ==(DateTime d1, DateTime d2)
```

Summary

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

Parameters

Parameter	Description
d1	The first `System.DateTime` to compare.
d2	The second `System.DateTime` to compare.

Return Value

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

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

```
[ILAsm]
.method public hidebysig static specialname bool op_GreaterThan(valuetype
System.DateTime t1, valuetype System.DateTime t2)

[C#]
public static bool operator >(DateTime t1, DateTime t2)
```

Summary

Returns a `System.Boolean` value indicating whether one specified `System.DateTime` is greater than another specified `System.DateTime`.

Parameters

Parameter	Description
t1	A `System.DateTime`.
t2	A `System.DateTime`.

Return Value

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

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

```
[ILAsm]
.method public hidebysig static specialname bool
op_GreaterThanOrEqual(valuetype System.DateTime t1, valuetype
System.DateTime t2)

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

Summary

Returns a `System.Boolean` value indicating whether one specified `System.DateTime` is greater than or equal to another specified `System.DateTime`.

Parameters

Parameter	Description
t1	A `System.DateTime`.
t2	A `System.DateTime`.

Return Value

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

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

```
[ILAsm]
.method public hidebysig static specialname bool op_Inequality(valuetype
System.DateTime d1, valuetype System.DateTime d2)

[C#]
public static bool operator !=(DateTime d1, DateTime d2)
```

Summary

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

Parameters

Parameter	Description
d1	A `System.DateTime`.
d2	A `System.DateTime`.

Return Value

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

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

```
[ILAsm]
.method public hidebysig static specialname bool op_LessThan(valuetype
System.DateTime t1, valuetype System.DateTime t2)

[C#]
public static bool operator <(DateTime t1, DateTime t2)
```

Summary

Returns a `System.Boolean` value indicating whether one specified `System.DateTime` is less than another specified `System.DateTime`.

Parameters

Parameter	Description
t1	A `System.DateTime`.
t2	A `System.DateTime`.

Return Value

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

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

```
[ILAsm]
.method public hidebysig static specialname bool
op_LessThanOrEqual(valuetype System.DateTime t1, valuetype System.DateTime
t2)

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

Summary

Returns a `System.Boolean` value indicating whether one specified `System.DateTime` is less than or equal to another specified `System.DateTime`.

Parameters

Parameter	Description
t1	A `System.DateTime`.
t2	A `System.DateTime`.

Return Value

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

DateTime.op_Subtraction(System.DateTime, System.TimeSpan) Method

```
[ILAsm]
.method public hidebysig static specialname valuetype System.DateTime
op_Subtraction(valuetype System.DateTime d, valuetype System.TimeSpan t)

[C#]
public static DateTime operator -(DateTime d, TimeSpan t)
```

Summary

Subtracts a specified `System.TimeSpan` from a specified `System.DateTime`.

Parameters

Parameter	Description
d	A `System.DateTime`.
t	A `System.TimeSpan`.

Return Value

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

Description

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

Exceptions

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

DateTime.op_Subtraction(System.DateTime, System.DateTime) Method

```
[ILAsm]
.method public hidebysig static specialname valuetype System.TimeSpan
op_Subtraction(valuetype System.DateTime d1, valuetype System.DateTime d2)

[C#]
public static TimeSpan operator -(DateTime d1, DateTime d2)
```

Summary

Subtracts a specified `System.DateTime` from another specified `System.DateTime` value, producing a time interval.

Parameters

Parameter	**Description**
d1	A `System.DateTime` (the minuend).
d2	A `System.DateTime` (the subtrahend).

Return Value

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

Description

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

Exceptions

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

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.

Exceptions

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.

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

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

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

Summary

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

Parameters

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

Return Value

The System.DateTime value obtained from *s*.

Description

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

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 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.

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.

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
s	A `System.String` containing the value to convert.
provider	A `System.IFormatProvider` that supplies a `System.Globalization.DateTimeFormatInfo` object containing culture-specific format information about *s*.
styles	One or more `System.Globalization.DateTimeStyles` values that specify the style of *s*. Specify multiple values for *styles* 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 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, and if the *styles* parameter is set to `System.Globalization.DateTimeStyles.NoCurrentDateDefault` the Gregorian year 1, month 1, day 1 are used. In all other cases where a date is not specified, the current date (`System.DateTime.Now`) is used.

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

For all settings of the *styles* parameter, 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.

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

```
[ILAsm]
.method public hidebysig static valuetype System.DateTime
ParseExact(string s, string format, class System.IFormatProvider provider,
valuetype System.Globalization.DateTimeStyles style)

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

Summary

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

Parameters

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

Return Value

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

Description

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

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.

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

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

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

Exceptions

Exception	**Condition**
System.ArgumentNullException	*s* or *format* is a null reference.
System.FormatException	*s* or format is an empty string. -or- *s* does not contain a date and time that were recognized as one of the patterns specified in *format.*

Example

This example demonstrates the `System.DateTime.ParseExact` method.

[C#]

```
using System;
using System.Globalization;

public class DateTimeTest {
 public static void Main() {
     DateTimeFormatInfo dtfi = new DateTimeFormatInfo();

     DateTime dt = DateTime.ParseExact(" January 22 ", dtfi.MonthDayPattern,
null, DateTimeStyles.AllowWhiteSpaces);
     Console.WriteLine(dt);
```

```
  }
}
```

The output is

```
1/22/2001 12:00:00 AM
```

DateTime.ParseExact(System.String, System.String[], System.IFormatProvider, System.Globalization.DateTimeStyles) Method

```
[ILAsm]
.method public hidebysig static valuetype System.DateTime
ParseExact(string s, string[] formats, class System.IFormatProvider
provider, valuetype System.Globalization.DateTimeStyles style)

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

Summary

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

Parameters

Parameter	Description
s	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.
formats	A `System.String` array containing the expected formats of *s*. [*Note:* For a list of valid *format* values, see `System.Globalization.DateTimeFormatInfo`.]
provider	A `System.IFormatProvider` that supplies a `System.Globalization.DateTimeFormatInfo` object containing culture-specific format information about *s*.
style	One or more `System.Globalization.DateTimeStyles` values that specify the style of *s*. Specify multiple values for *styles* using the bitwise OR operator.

Return Value

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

Description

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

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.

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

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

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

Exceptions

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

Example

This example demonstrates the `System.DateTime.ParseExact` method.

[C#]

```
using System;
using System.Globalization;

public class DateTimeTest {
 public static void Main() {
     DateTimeFormatInfo dtfi = new DateTimeFormatInfo();
     string [] patterns = {dtfi.LongTimePattern, dtfi.ShortTimePattern};

     DateTime dt = DateTime.ParseExact("10:11:12", patterns, null,
DateTimeStyles.NoCurrentDateDefault);
     Console.WriteLine(dt);
 }
}
```

The output is

```
1/1/0001 10:11:12 AM
```

DateTime.ParseExact(System.String, System.String, System.IFormatProvider) Method

```
[ILAsm]
.method public hidebysig static valuetype System.DateTime
ParseExact(string s, string format, class System.IFormatProvider provider)

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

Summary

Converts the specified `System.String` representation of a date and time to its `System.DateTime` equivalent using a specified format and `System.IFormatProvider`.

Parameters

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

Return Value

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

Description

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

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.

If the *s* 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.

Leading, trailing, and inner white space characters are not allowed.

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

Exceptions

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

Example

This example demonstrates the `System.DateTime.ParseExact` method.

[C#]

```
using System;
using System.Globalization;

public class DateTimeTest {
 public static void Main() {
     DateTimeFormatInfo dtfi = new DateTimeFormatInfo();

     DateTime dt = DateTime.ParseExact("January 22", dtfi.MonthDayPattern,
null);
     Console.WriteLine(dt);
 }
}
```

The output is

```
1/22/2001 12:00:00 AM
```

DateTime.Subtract(System.TimeSpan) Method

```
[ILAsm]
.method public hidebysig instance valuetype System.DateTime
Subtract(valuetype System.TimeSpan value)

[C#]
public DateTime Subtract(TimeSpan value)
```

Summary

Subtracts a specified `System.TimeSpan` from the current instance.

Parameters

Parameter	Description
value	An instance of `System.TimeSpan`.

Return Value

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

Exceptions

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

DateTime.Subtract(System.DateTime) Method

```
[ILAsm]
.method public hidebysig instance valuetype System.TimeSpan
Subtract(valuetype System.DateTime value)

[C#]
public TimeSpan Subtract(DateTime value)
```

Summary

Subtracts a specified date and time from the current instance.

Parameters

Parameter	Description
value	An instance of `System.DateTime`.

Return Value

A `System.TimeSpan` interval equal to the date and time represented by the current instance minus the date and time represented by the specified `System.DateTime`.

DateTime.ToLocalTime() Method

```
[ILAsm]
.method public hidebysig instance valuetype System.DateTime ToLocalTime()

[C#]
public DateTime ToLocalTime()
```

Summary

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

Return Value

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

Description

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

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

DateTime.ToLongDateString() Method

[ILAsm]
```
.method public hidebysig instance string ToLongDateString()
```

[C#]
```
public string ToLongDateString()
```

Summary

Converts the date denoted by the current instance to its equivalent long date `System.String` representation.

Return Value

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

Description

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

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

For more information regarding the long date specifier, see `System.Globalization.DateTimeFormatInfo`.

]

DateTime.ToLongTimeString() Method

[ILAsm]
```
.method public hidebysig instance string ToLongTimeString()
```

[C#]
```
public string ToLongTimeString()
```

Summary

Converts the time denoted by the current instance to its equivalent long time `System.String` representation.

Return Value

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

Description

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

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

For more information regarding the long time specifier, see `System.Globalization.DateTimeFormatInfo`.

]

DateTime.ToShortDateString() Method

[ILAsm]
```
.method public hidebysig instance string ToShortDateString()
```

[C#]
```
public string ToShortDateString()
```

Summary

Converts the date denoted by the current instance to its equivalent short date `System.String` representation.

Return Value

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

Description

The value of the current instance is formatted using the short date format specifier, 'd'.

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

For more information regarding the short date specifier, see `System.Globalization.DateTimeFormatInfo`.

]

DateTime.ToShortTimeString() Method

```
[ILAsm]
.method public hidebysig instance string ToShortTimeString()

[C#]
public string ToShortTimeString()
```

Summary

Converts the time denoted by this instance to its equivalent short time `System.String` representation.

Return Value

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

Description

The value of the current instance is formatted using the short time format specifier, 't'.

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

For more information regarding the short time specifier, see `System.Globalization.DateTimeFormatInfo`.

]

DateTime.ToString(System.String) Method

```
[ILAsm]
.method public hidebysig instance string ToString(string format)

[C#]
public string ToString(string format)
```

Summary

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

Parameters

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

Return Value

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

Description

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

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

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

For information on formatting system-supplied data types, see the `System.IFormattable` interface.

]

Exceptions

Exception	Condition
System.FormatException	The length of the *format* string is 1, and it is not one of the format specifier characters defined for `System.Globalization.DateTimeFormatInfo`. -or-

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

DateTime.ToString() Method

[ILAsm]
```
.method public hidebysig virtual string ToString()
```

[C#]
```
public override string ToString()
```

Summary

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

Return Value

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

Description

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

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

This method overrides `System.Object.ToString`.

]

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

```
[ILAsm]
.method public final hidebysig virtual string ToString(string format,
class System.IFormatProvider provider)

[C#]
public string ToString(string format, IFormatProvider provider)
```

Summary

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

Parameters

Parameter	Description
format	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`.]
provider	A `System.IFormatProvider` that supplies a `System.Globalization.DateTimeFormatInfo` instance containing culture-specific formatting information.

Return Value

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

Description

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

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

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

This method is implemented to support the `System.IFormattable` interface.

]

Exceptions

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

DateTime.ToString(System.IFormatProvider) Method

```
[ILAsm]
.method public final hidebysig virtual string ToString(class
System.IFormatProvider provider)

[C#]
public string ToString(IFormatProvider provider)
```

Summary

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

Parameters

Parameter	Description
provider	A `System.IFormatProvider` that supplies a `System.Globalization.DateTimeFormatInfo` containing culture-specific formatting information.

Return Value

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

Description

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

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

[*Note:* The general format specifier ("G") provides the general date pattern including the long time form, equivalent to `System.Globalization.DateTimeFormatInfo.ShortDatePattern` combined with `System.Globalization.DateTimeFormatInfo.LongTimePattern`. For more information on format specifiers, see `System.Globalization.DateTimeFormatInfo`. For information on formatting system-supplied data types, see the `System.IFormattable` interface.

]

DateTime.ToUniversalTime() Method

[ILAsm]
```
.method public hidebysig instance valuetype System.DateTime
ToUniversalTime()
```

[C#]
```
public DateTime ToUniversalTime()
```

Summary

Converts the current `System.DateTime` value to coordinated universal time (UTC).

Return Value

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

Description

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

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

DateTime.Date Property

```
[ILAsm]
.property valuetype System.DateTime Date { public hidebysig specialname
instance valuetype System.DateTime get_Date() }

[C#]
public DateTime Date { get; }
```

Summary

Gets the date component of the current instance.

Property Value

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

Description

This property is read-only.

DateTime.Day Property

[ILAsm]
```
.property int32 Day { public hidebysig specialname instance int32
get_Day() }
```

[C#]
```
public int Day { get; }
```

Summary

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

Property Value

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

Description

This property is read-only.

DateTime.DayOfYear Property

[ILAsm]
```
.property int32 DayOfYear { public hidebysig specialname instance int32
get_DayOfYear() }
```

[C#]
```
public int DayOfYear { get; }
```

Summary

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

Property Value

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

Description

This property is read-only.

DateTime.Hour Property

[ILAsm]
```
.property int32 Hour { public hidebysig specialname instance int32
get_Hour() }
```

[C#]
```
public int Hour { get; }
```

Summary

Gets the hour represented by the current instance.

Property Value

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

Description

This property is read-only.

DateTime.Millisecond Property

[ILAsm]
```
.property int32 Millisecond { public hidebysig specialname instance int32
get_Millisecond() }
```

[C#]
```
public int Millisecond { get; }
```

Summary

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

Property Value

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

Description

This property is read-only.

DateTime.Minute Property

[ILAsm]
```
.property int32 Minute { public hidebysig specialname instance int32
get_Minute() }
```

[C#]
```
public int Minute { get; }
```

Summary

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

Property Value

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

Description

This property is read-only.

DateTime.Month Property

```
[ILAsm]
.property int32 Month { public hidebysig specialname instance int32
get_Month() }

[C#]
public int Month { get; }
```

Summary

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

Property Value

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

Description

This property is read-only.

DateTime.Now Property

[ILAsm]
```
.property valuetype System.DateTime Now { public hidebysig static
specialname valuetype System.DateTime get_Now() }
```

[C#]
```
public static DateTime Now { get; }
```

Summary

Gets a `System.DateTime` representing the current local date and time.

Description

The resolution of this property depends on the system timer.

This property is read-only.

DateTime.Second Property

[ILAsm]
```
.property int32 Second { public hidebysig specialname instance int32
get_Second() }
```

[C#]
```
public int Second { get; }
```

Summary

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

Property Value

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

Description

This property is read-only.

DateTime.Ticks Property

```
[ILAsm]
.property int64 Ticks { public hidebysig specialname instance int64
get_Ticks() }

[C#]
public long Ticks { get; }
```

Summary

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

Property Value

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

Description

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

This property is read-only.

DateTime.TimeOfDay Property

[ILAsm]
```
.property valuetype System.TimeSpan TimeOfDay { public hidebysig
specialname instance valuetype System.TimeSpan get_TimeOfDay() }
```

[C#]
```
public TimeSpan TimeOfDay { get; }
```

Summary

Gets the time of day of the current instance.

Property Value

A `System.TimeSpan` instance set to the time component of the current instance.

Description

This property is read-only.

DateTime.Today Property

[ILAsm]
```
.property valuetype System.DateTime Today { public hidebysig static
specialname valuetype System.DateTime get_Today() }
```

[C#]
```
public static DateTime Today { get; }
```

Summary

Gets the current date.

Property Value

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

Description

This property is read-only.

DateTime.UtcNow Property

[ILAsm]
```
.property valuetype System.DateTime UtcNow { public hidebysig static
specialname valuetype System.DateTime get_UtcNow() }
```

[C#]
```
public static DateTime UtcNow { get; }
```

Summary

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

Property Value

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

Description

This property is read-only.

DateTime.Year Property

[ILAsm]
```
.property int32 Year { public hidebysig specialname instance int32
get_Year() }
```

[C#]
```
public int Year { get; }
```

Summary

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

Property Value

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

Description

This property is read-only.