**C# - Struttura DateTime**

Rappresenta un istante di tempo, in genere espresso come data e ora del giorno.

Il tipo di valore DateTime rappresenta i valori di data e ora compresi tra le ore 0:00:00 del 1° gennaio 0001 D.C. e le ore 23.59.59 del 31 dicembre 9999 D.C.

I valori di ora vengono misurati in unità di 100 nanosecondi, definite segni di graduazione. Una determinata data è rappresentata dal numero di segni di graduazione a partire dalle ore 0:00 del 1° gennaio 0001 D.C. del calendario Gregoriano.

Il tipo DateTime espone i seguenti membri:

**[http://i.msdn.microsoft.com/Hash/030c41d9079671d09a62d8e2c1db6973.gif](javascript:void(0))Costruttori**

|  |  |
| --- | --- |
| **Nome** | **Descrizione** |
| [DateTime(Int64)](http://msdn.microsoft.com/it-it/library/z2xf7zzk.aspx) | Inizializza una nuova istanza della struttura DateTime su un numero di segni di graduazione specificato. |
| [DateTime(Int32, Int32, Int32)](http://msdn.microsoft.com/it-it/library/xcfzdy4x.aspx) | Inizializza una nuova istanza della struttura DateTime su anno, mese e giorno specificati. |
| [DateTime(Int32, Int32, Int32, Int32, Int32, Int32)](http://msdn.microsoft.com/it-it/library/272ba130.aspx) | Inizializza una nuova istanza della struttura DateTime su anno, mese, giorno, ora, minuto e secondo specificati. |
| [DateTime(Int32, Int32, Int32, Int32, Int32, Int32, Int32)](http://msdn.microsoft.com/it-it/library/wb249tb7.aspx) | Inizializza una nuova istanza della struttura DateTime su anno, mese, giorno, ora, minuto, secondo e millisecondo specificati. |
|  |  |
|  |  |

**[http://i.msdn.microsoft.com/Hash/030c41d9079671d09a62d8e2c1db6973.gif](javascript:void(0))Proprietà**

|  |  |
| --- | --- |
| **Nome** | **Descrizione** |
| [Date](http://msdn.microsoft.com/it-it/library/system.datetime.date.aspx) | Ottiene il componente di data dell'istanza. |
| [Day](http://msdn.microsoft.com/it-it/library/system.datetime.day.aspx) | Ottiene il giorno del mese rappresentato dall'istanza. |
| [DayOfWeek](http://msdn.microsoft.com/it-it/library/system.datetime.dayofweek.aspx) | Una costante enumerata che indica il giorno della settimana rappresentato dall'istanza. Il valore delle costanti nell'enumerazione DayOfWeek è compreso tra DayOfWeek.Sunday e DayOfWeek.Saturday. Se si esegue il cast in un intero, tale valore sarà compreso tra zero (che indica DayOfWeek.Sunday) e sei (che indica DayOfWeek.Saturday). Non riflette le impostazioni internazionali e della lingua di un sistema. |
| [DayOfYear](http://msdn.microsoft.com/it-it/library/system.datetime.dayofyear.aspx) | Ottiene il giorno dell'anno rappresentato dall'istanza. |
| [Hour](http://msdn.microsoft.com/it-it/library/system.datetime.hour.aspx) | Ottiene il componente di ora della data rappresentata dall'istanza. |
| [Millisecond](http://msdn.microsoft.com/it-it/library/system.datetime.millisecond.aspx) | Ottiene il componente di millisecondi della data rappresentata dall'istanza. |
| [Minute](http://msdn.microsoft.com/it-it/library/system.datetime.minute.aspx) | Ottiene il componente di minuti della data rappresentata dall'istanza. |
| [Month](http://msdn.microsoft.com/it-it/library/system.datetime.month.aspx) | Ottiene il componente di mese della data rappresentata dall'istanza. |
| [Now](http://msdn.microsoft.com/it-it/library/system.datetime.now.aspx) | Ottiene un oggetto DateTime impostato su data e ora correnti sul computer, espresso come ora locale. |
| [Second](http://msdn.microsoft.com/it-it/library/system.datetime.second.aspx) | Ottiene il componente di secondi della data rappresentata dall'istanza. |
| [Ticks](http://msdn.microsoft.com/it-it/library/system.datetime.ticks.aspx) | Ottiene il numero di segni di graduazione che rappresentano data e ora dell'istanza. |
| [TimeOfDay](http://msdn.microsoft.com/it-it/library/system.datetime.timeofday.aspx) | Ottiene l'ora del giorno per l'istanza. |
| [Today](http://msdn.microsoft.com/it-it/library/system.datetime.today.aspx) | Ottiene la data corrente. |
| [Year](http://msdn.microsoft.com/it-it/library/system.datetime.year.aspx) | Ottiene il componente dell'anno della data rappresentata dall'istanza. |

**[http://i.msdn.microsoft.com/Hash/030c41d9079671d09a62d8e2c1db6973.gif](javascript:void(0))Metodi**

|  |  |
| --- | --- |
| **Nome** | **Descrizione** |
| [Add](http://msdn.microsoft.com/it-it/library/system.datetime.add.aspx) | Restituisce un nuovo oggetto DateTime che aggiunge il valore dell'oggetto [TimeSpan](http://msdn.microsoft.com/it-it/library/system.timespan.aspx) specificato al valore di questa istanza. |
| [AddDays](http://msdn.microsoft.com/it-it/library/system.datetime.adddays.aspx) | Restituisce un nuovo oggetto DateTime che aggiunge il numero specificato di giorni al valore di questa istanza. |
| [AddHours](http://msdn.microsoft.com/it-it/library/system.datetime.addhours.aspx) | Restituisce un nuovo oggetto DateTime che aggiunge il numero specificato di ore al valore di questa istanza. |
| [AddMilliseconds](http://msdn.microsoft.com/it-it/library/system.datetime.addmilliseconds.aspx) | Restituisce un nuovo oggetto DateTime che aggiunge il numero specificato di millisecondi al valore di questa istanza. |
| [AddMinutes](http://msdn.microsoft.com/it-it/library/system.datetime.addminutes.aspx) | Restituisce un nuovo oggetto DateTime che aggiunge il numero specificato di minuti al valore di questa istanza. |
| [AddMonths](http://msdn.microsoft.com/it-it/library/system.datetime.addmonths.aspx) | Restituisce un nuovo oggetto DateTime che aggiunge il numero specificato di mesi al valore di questa istanza. |
| [AddSeconds](http://msdn.microsoft.com/it-it/library/system.datetime.addseconds.aspx) | Restituisce un nuovo oggetto DateTime che aggiunge il numero specificato di secondi al valore di questa istanza. |
| [AddTicks](http://msdn.microsoft.com/it-it/library/system.datetime.addticks.aspx) | Restituisce un nuovo oggetto DateTime che aggiunge il numero specificato di segni di graduazione al valore di questa istanza. |
| [AddYears](http://msdn.microsoft.com/it-it/library/system.datetime.addyears.aspx) | Restituisce un nuovo oggetto DateTime che aggiunge il numero specificato di anni al valore di questa istanza. |
| [Compare](http://msdn.microsoft.com/it-it/library/system.datetime.compare.aspx) | Confronta due istanze di DateTime e restituisce un intero che indica se la prima istanza è precedente, uguale o successiva alla seconda istanza. |
| [CompareTo(DateTime)](http://msdn.microsoft.com/it-it/library/5ata5aya.aspx) | Confronta il valore di questa istanza con un valore DateTime specificato e restituisce un intero che indica se l'istanza è precedente, uguale o successiva al valore DateTime specificato. |
| [DaysInMonth](http://msdn.microsoft.com/it-it/library/system.datetime.daysinmonth.aspx) | Restituisce il numero di giorni nel mese e anno specificati. |
| [Equals(DateTime)](http://msdn.microsoft.com/it-it/library/635d5466.aspx) | Restituisce un valore che indica se l'istanza è uguale all'istanza dell'oggetto DateTime specificata. |
| [Equals(DateTime, DateTime)](http://msdn.microsoft.com/it-it/library/wk5t802s.aspx) | Restituisce un valore che indica se due istanze dell'oggetto DateTime sono uguali. |
| [IsLeapYear](http://msdn.microsoft.com/it-it/library/system.datetime.isleapyear.aspx) | Restituisce un'indicazione che informa se l'anno specificato è bisestile. |
| [Parse(String)](http://msdn.microsoft.com/it-it/library/1k1skd40.aspx) | Converte una determinata rappresentazione di stringa di una data e ora nell'oggetto DateTime equivalente. |
| [Subtract(DateTime)](http://msdn.microsoft.com/it-it/library/8ysw4sby.aspx) | Sottrae la data e l'ora specificate dall'istanza. |
| [Subtract(TimeSpan)](http://msdn.microsoft.com/it-it/library/ae6246z1.aspx) | Sottrae la durata specificata dall'istanza. |
| [ToString](http://msdn.microsoft.com/it-it/library/k494fzbf.aspx) | Converte il valore dell'oggetto DateTime corrente nella rappresentazione in forma di stringa equivalente. |

**[http://i.msdn.microsoft.com/Hash/030c41d9079671d09a62d8e2c1db6973.gif](javascript:void(0))Operatori**

|  |  |
| --- | --- |
| **Nome** | **Descrizione** |
| [Addition](http://msdn.microsoft.com/it-it/library/system.datetime.op_addition.aspx) | Aggiunge un determinato intervallo di tempo a una data e a un'ora specificate, producendo una data e un'ora nuove. |
| [Equality](http://msdn.microsoft.com/it-it/library/system.datetime.op_equality.aspx) | Determina se due istanze specificate dell'oggetto DateTime sono uguali. |
| [GreaterThan](http://msdn.microsoft.com/it-it/library/system.datetime.op_greaterthan.aspx) | Determina se un DateTime specificato è maggiore di un altro DateTime specificato. |
| [GreaterThanOrEqual](http://msdn.microsoft.com/it-it/library/system.datetime.op_greaterthanorequal.aspx) | Determina se un DateTime specificato è maggiore o uguale a un altro DateTime specificato. |
| [Inequality](http://msdn.microsoft.com/it-it/library/system.datetime.op_inequality.aspx) | Determina se due istanze specificate dell'oggetto DateTime non sono uguali. |
| [LessThan](http://msdn.microsoft.com/it-it/library/system.datetime.op_lessthan.aspx) | Determina se un DateTime specificato è minore di un altro DateTime specificato. |
| [LessThanOrEqual](http://msdn.microsoft.com/it-it/library/system.datetime.op_lessthanorequal.aspx) | Determina se un DateTime specificato è minore o uguale a un altro DateTime specificato. |
| [Subtraction(DateTime, DateTime)](http://msdn.microsoft.com/it-it/library/1905yhe2.aspx) | Sottrae una data e un'ora specificate da un'altra data e un'altra ora specificate e restituisce un intervallo di tempo. |
| [Subtraction(DateTime, TimeSpan)](http://msdn.microsoft.com/it-it/library/s85h2eca.aspx) | Sottrae un intervallo di tempo specificato da una data e un'ora specificate e restituisce una data e un'ora nuove. |

**[http://i.msdn.microsoft.com/Hash/030c41d9079671d09a62d8e2c1db6973.gif](javascript:void(0))Campi**

|  |  |
| --- | --- |
| **Nome** | **Descrizione** |
| [MaxValue](http://msdn.microsoft.com/it-it/library/system.datetime.maxvalue.aspx) | Rappresenta il valore massimo possibile dell'oggetto DateTime. Questo è un campo di sola lettura. |
| [MinValue](http://msdn.microsoft.com/it-it/library/system.datetime.minvalue.aspx) | Rappresenta il valore più piccolo possibile di DateTime. Questo è un campo di sola lettura. |

**Creazione di un'istanza di un oggetto DateTime**

È possibile creare un nuovo valore DateTime in uno dei seguenti modi:

Chiamando uno degli overload del costruttore DateTime che consentono di specificare determinati elementi del valore di data e ora (ad esempio l'anno, il mese e il giorno o il numero di tick). Nell'istruzione seguente viene illustrata una chiamata a uno dei costruttori DateTime per creare una data con un anno, un mese, un giorno, un'ora, un minuto e un secondo specifici.

DateTime date1 = new DateTime(2008, 5, 1, 8, 30, 52);

Assegnando all'oggetto DateTime un valore di data e ora restituito da una proprietà o un metodo. Nell'esempio seguente vengono assegnate la data e l'ora correnti, la data e l'ora UTC (Coordinated Universal Time) correnti e la data corrente a tre nuove variabili

DateTime date1 = DateTime.Now;

DateTime date2 = DateTime.Today;

Analizzando la rappresentazione di stringa di un valore di data e ora. Il metodo Parse converte una stringa nel valore di data e ora equivalente. Nell'esempio seguente viene utilizzato il metodo Parse per analizzare una stringa e convertirla in un valore DateTime.

string dateString = "5/1/2008 8:30:52 AM";

DateTime date1 = DateTime.Parse(dateString);

**Valori DateTime e relative rappresentazioni di stringa**

Internamente, tutti i valori DateTime sono rappresentati come numero di cicli (il numero di intervalli da 100 nanosecondi) che sono trascorsi dalle ore 12:00:00, mezzanotte, del 1° gennaio 0001. Il valore DateTime effettivo è indipendente dalla modalità in cui viene visualizzato tale valore se visualizzato in un elemento dell'interfaccia utente o in caso di scrittura in un file. L'aspetto di un valore DateTime è il risultato di un'operazione di formattazione, vale a dire di conversione del valore nella relativa rappresentazione di stringa.

Dal momento che l'aspetto dei valori di data e ora dipende da fattori quali le impostazioni cultura, gli standard internazionali, i requisiti dell'applicazione e le preferenze personali, la struttura DateTime consente una grande flessibilità nella formattazione dei valori di data e ora tramite gli overload del metodo ToString. Il metodo DateTime.ToString predefinito restituisce la rappresentazione di stringa di un valore di data e ora utilizzando il modello di data breve e ora estesa delle impostazioni cultura correnti. Nell'esempio seguente viene utilizzato il metodo DateTime.ToString predefinito per visualizzare la data e l'ora utilizzando il modello di data breve e ora estesa delle impostazioni cultura en-US, ovvero le impostazioni cultura correnti del computer sul quale è stato eseguito l'esempio.

DateTime date1 = new DateTime(2008, 3, 1, 7, 0, 0);

Console.WriteLine(date1.ToString());

// For en-US culture, displays 3/1/2008 7:00:00 AM

**Operazioni sulle strutture DateTime**

I calcoli effettuati sulle strutture DateTime, ad esempio mediante i metodi Add o Subtract, non modificano i valori delle strutture. I calcoli di questo tipo, invece, restituiscono una nuova struttura DateTime il cui valore è il risultato del calcolo stesso**.**

Per eseguire le proprie operazioni, i membri della struttura DateTime utilizzano in modo implicito il calendario gregoriano.

Nelle operazioni eseguite dai membri del tipo DateTime si prendono in considerazione dettagli quali gli anni bisestili e il numero di giorni in un mese.

**DateTime rispetto aTimeSpan**

I tipi di valore DateTime e TimeSpan differiscono poiché DateTime rappresenta un istante di tempo, mentre TimeSpan rappresenta un intervallo di tempo. Ciò significa, ad esempio, che è possibile sottrarre un'istanza di DateTime da un'altra istanza per ottenere un oggetto TimeSpan rappresentante l'intervallo di tempo compreso fra le due. Oppure è possibile aggiungere un oggetto TimeSpan positivo all'oggetto DateTime corrente per ottenere un valore DateTime che rappresenta una data futura.

A un oggetto DateTime è possibile aggiungere o sottrarre un intervallo di tempo. Gli intervalli di tempo possono essere negativi o positivi, possono essere espressi in unità, ad esempio segni di graduazione o secondi, oppure come un oggetto TimeSpan.

**Esempi:**

**Proprietà e Campi:**

Console.WriteLine(DateTime.MaxValue);

Console.WriteLine(DateTime.MinValue);

Console.WriteLine(DateTime.Now);

Console.WriteLine(DateTime.Today);

Console.WriteLine();

DateTime d1 = DateTime.Now;

Console.WriteLine(d1.Date);

Console.WriteLine(d1.Day);

Console.WriteLine(d1.DayOfWeek);

Console.WriteLine((int) d1.DayOfWeek);

Console.WriteLine(d1.DayOfYear);

Console.WriteLine(d1.Hour);

Console.WriteLine(d1.Millisecond);

Console.WriteLine(d1.Minute);

Console.WriteLine(d1.Month);

Console.WriteLine(d1.Second);

Console.WriteLine(d1.Ticks);

Console.WriteLine(d1.TimeOfDay);

Console.WriteLine(d1.Year);

//Output

31/12/9999 23:59:59

01/01/0001 00:00:00

28/02/2012 11:50:06

28/02/2012 00:00:00

28/02/2012 00:00:00

28

Tuesday

2

59

11

961

50

2

6

634660266069615413

11:50:06.9615413

2012

**Metodi / Calcolo:**

**Metodi Add…**

DateTime d1 = DateTime.Now;

DateTime d2 = new DateTime(2012, 02, 18);

Console.WriteLine("Data1: {0}", d1);

Console.WriteLine("Data2: {0}", d2);

Console.WriteLine();

Console.WriteLine(d1.AddSeconds(20));

Console.WriteLine(d1.AddSeconds(-20));

Console.WriteLine(d1.AddHours(2));

Console.WriteLine(d1.AddHours(2).AddSeconds(2));

Console.WriteLine("Ieri: {0}", DateTime.Today.AddDays(-1));

Console.WriteLine("Domani: {0}", DateTime.Today.AddDays(1));

//Output

Data1: 29/02/2012 11:32:40

Data2: 18/02/2012 00:00:00

29/02/2012 11:33:00

29/02/2012 11:32:20

29/02/2012 13:32:40

29/02/2012 13:32:42

Ieri: 28/02/2012 00:00:00

Domani: 01/03/2012 00:00:00

**Metodi Compare, CompareTo**

DateTime d1 = DateTime.Now;

DateTime d2 = d1.AddSeconds(2);

DateTime d3 = d1.AddSeconds(-2);

DateTime d4 = d1;

Console.WriteLine(d1);

Console.WriteLine();

Console.WriteLine(d2);

Console.WriteLine(d3);

Console.WriteLine(d4);

Console.WriteLine();

Console.WriteLine(DateTime.Compare(d1, d2));

Console.WriteLine(DateTime.Compare(d1, d3));

Console.WriteLine(DateTime.Compare(d1, d4));

Console.WriteLine();

Console.WriteLine(d1.CompareTo(d2));

Console.WriteLine(d1.CompareTo(d3));

Console.WriteLine(d1.CompareTo(d4));

//Output

28/02/2012 11:28:24

28/02/2012 11:28:26

28/02/2012 11:28:22

28/02/2012 11:28:24

-1 //Il primo è più piccolo

1 //Il primo è più grande

0 //Sono uguali

-1 //Il primo è più piccolo

1 //Il primo è più grande

0 //Sono uguali

**Metodo DaysInMonth**

DateTime d1 = DateTime.Now;

int marzo = 3;

int febbraio = 2;

int giornimarzo = System.DateTime.DaysInMonth(d1.Year, marzo);

int giornifebbraio = System.DateTime.DaysInMonth(d1.Year, febbraio);

int giornifebbraio2 = System.DateTime.DaysInMonth(2011,febbraio);

Console.WriteLine(d1);

Console.WriteLine(giornimarzo);

Console.WriteLine(giornifebbraio2);

Console.WriteLine(giornifebbraio);

//Output

28/02/2012 11:28:24

31 //Sono i giorni di Marzo

28 //Nel 2011 Febbraio aveva 28 giorni

29 //Essendo il 2012 bisestile, Febbraio ha 29 giorni

**Metodo Equals**

DateTime d1 = DateTime.Now;

DateTime d2 = d1.AddSeconds(2);

DateTime d3 = d1;

Console.WriteLine(d1);

Console.WriteLine();

Console.WriteLine(d2);

Console.WriteLine(d3);

Console.WriteLine();

Console.WriteLine(d1.Equals(d2));

Console.WriteLine(d1.Equals(d3));

Console.WriteLine();

Console.WriteLine(Equals(d1, d2));

Console.WriteLine(Equals(d1, d3));

//Output

28/02/2012 11:28:24

28/02/2012 11:28:26

28/02/2012 11:28:24

False

True

False

True

**Metodo IsLeapYear**

DateTime d1 = DateTime.Now;

Console.WriteLine(d1);

Console.WriteLine();

Console.WriteLine(DateTime.IsLeapYear(d1.Year));

d1 = d1.AddYears(1);

Console.WriteLine(DateTime.IsLeapYear(d1.Year));

//Output

28/02/2012 11:28:24

True //2012 bisestile

False //2013 non bisestile

**Metodo Parse**

DateTime d1 = DateTime.Now;

string s1 = "28/02/2012 11:28:24";

string s2 = "2012/02/28";

string s3 = "2012/02/28 11:13:02";

string s4 = "2012-02-28 11:13:02";

d1 = DateTime.Parse(s1);

Console.WriteLine(d1);

d1 = DateTime.Parse(s2);

Console.WriteLine(d1);

d1 = DateTime.Parse(s3);

Console.WriteLine(d1);

d1 = DateTime.Parse(s4);

Console.WriteLine(d1);

//Output

28/02/2012 11:28:24

28/02/2012 00:00:00

28/02/2012 11:13:02

28/02/2012 11:13:02

**Metodo Subtract**

DateTime d1 = DateTime.Now;

DateTime d2 = d1.AddSeconds(2);

DateTime d3 = d1.AddSeconds(-2);

DateTime d4 = new DateTime(2012, 03, 02, 18, 34, 15);

DateTime d5 = new DateTime(2012, 03, 02, 10, 20, 20);

TimeSpan ts = d4.Subtract(d5);

Console.WriteLine(d1);

Console.WriteLine();

Console.WriteLine(d2);

Console.WriteLine(d3);

Console.WriteLine(d4);

Console.WriteLine(d5);

Console.WriteLine(ts);

Console.WriteLine();

Console.WriteLine(d1.Subtract(d2));

Console.WriteLine(d1.Subtract(d3));

Console.WriteLine(d1.Subtract(d4));

Console.WriteLine(d1.Subtract(d5));

Console.WriteLine(d1.Subtract(ts));

//Output

02/03/2012 18:04:27

02/03/2012 18:04:29

02/03/2012 18:04:25

02/03/2012 18:34:15

02/03/2012 10:20:20

00:13:55

-00:00:02

00:00:02

-00:29:47.1069604

07:44:07.8930396

02/03/2012 09:50:32

02/03/2012 18:04:29

**Metodo ToString**

DateTime d1 = DateTime.Now;

Console.WriteLine(d1.ToString());

string s = Convert.ToString(d1);

Console.WriteLine(s);

//Output

05/03/2012 17:03:31

05/03/2012 17:03:31

**Operatori:**

**Operatore Somma**

DateTime dTime = new DateTime(1980, 8, 5);

// tSpan is 17 days, 4 hours, 2 minutes and 1 second.

TimeSpan tSpan = new TimeSpan(17, 4, 2, 1);

// Result gets 8/22/1980 4:02:01 AM.

DateTime result = dTime + tSpan;

Console.WriteLine("Data Iniziale \t{0} \nIntervallo \t{1} \nData Finale \t{2}", dTime, tSpan, result);

//Output

Data Iniziale 05/08/1980 00:00:00

Intervallo 17.04:02:01

Data Finale 22/08/1980 04:02:01

Errore: DateTime result = dTime + dTime;

**Operatore Sottrazione**

DateTime dTime1 = new DateTime(1980, 8, 5, 5, 7, 9);

DateTime dTime2 = new DateTime(1980, 8, 9, 5, 7, 10);

// tSpan is 4 days, 0 hours, 0 minutes and 1 second.

TimeSpan tSpan = new TimeSpan(4, 0, 0, 1);

Console.WriteLine("Data 1 {0}", dTime1);

Console.WriteLine("Data 2 {0}", dTime2);

Console.WriteLine("Data 2 - Data 1 {0}", dTime2 - dTime1);

Console.WriteLine("Intervallo {0}", tSpan);

Console.WriteLine("Data 2 - Intervallo {0}", dTime2 - tSpan);

//Output

Data 1 05/08/1980 05:07:09

Data 2 09/08/1980 05:07:10

Data 2 - Data 1 4.00:00:01

Intervallo 4.00:00:01

Data 2 - Intervallo 05/08/1980 05:07:09

**Operatori Logici**

DateTime dTime1 = new DateTime(1980, 8, 5);

DateTime dTime2 = new DateTime(1980, 11, 18);

Console.WriteLine("Data 1 {0} \nData 2 {1}", dTime1, dTime2);

Console.WriteLine();

Console.WriteLine("Data 1 = Data 2: {0}", dTime1 == dTime2);

Console.WriteLine("Data 1 # Data 2: {0}", dTime1 != dTime2);

Console.WriteLine("Data 1 > Data 2: {0}", dTime1 > dTime2);

Console.WriteLine("Data 1 >= Data 2: {0}", dTime1 >= dTime2);

Console.WriteLine("Data 1 < Data 2: {0}", dTime1 < dTime2);

Console.WriteLine("Data 1 <= Data 2: {0}", dTime1 <= dTime2);

//Output

Data 1 05/08/1980 00:00:00

Data 2 18/11/1980 00:00:00

Data 1 = Data 2: False

Data 1 # Data 2: True

Data 1 > Data 2: False

Data 1 >= Data 2: False

Data 1 < Data 2: True

Data 1 <= Data 2: True

**Esempi di codice:**

**For e DateTime**

* Contare il numero di lunedì in un determinato intervallo di date

DateTime startDate = DateTime.Now;

DateTime endDate = DateTime.Now.AddDays(27);

int mondayCount = 0;

for (DateTime dt = startDate; dt < endDate; dt = dt.AddDays(1.0))

{

if (dt.DayOfWeek == DayOfWeek.Monday)

{ mondayCount++; }

}

Console.WriteLine(mondayCount);

**DayOfWeek**

* Utilizzo di DayOfWeek

DateTime dt = new DateTime(2014, 1, 12);

Console.WriteLine(dt.DayOfWeek == DayOfWeek.Monday);

Console.WriteLine(dt.DayOfWeek.ToString() == "Tuesday");

Console.WriteLine((int) dt.DayOfWeek == 0); //Domenica