

Curso C# Completo Programação Orientada a Objetos + Projetos

Capítulo: Tópicos especiais em C# - PARTE 1

<http://educandoweb.com.br>

Prof. Dr. Nelio Alves

Inferência de tipos: palavra var

<http://educandoweb.com.br>

Prof. Dr. Nelio Alves

Demo

```
var x = 10;  
var y = 20.0;  
var z = "Maria";
```

```
Console.WriteLine(x);  
Console.WriteLine(y);  
Console.WriteLine(z);
```

var = o tipo da variável é inferido confor a 1ª inicialização dessa variável

- A mesma só deve ser sempre usada em substituição das variáveis normais.
- Mas sim, quando não temos a certeza do tipo de objeto que poderá ser retornado

Aula 86

Sintaxe alternativa: switch-case

<http://educandoweb.com.br>

Prof. Dr. Nelio Alves

switch-case

Estrutura opcional a vários if-else encadeados, **quando a condição envolve o teste do valor de uma variável.**

Sintaxe:

```
var minhaVariavel = (...);

switch (minhaVariavel) {
    case 1:
        Console.WriteLine("Caso 1");
        break;
    case 2:
        Console.WriteLine("Caso 2");
        break;
    default:
        Console.WriteLine("Caso padrão");
        break;
}
```

else if

```
int x = int.Parse(Console.ReadLine());
string day;

if (x == 1) {
    day = "Sunday";
}
else if (x == 2) {
    day = "Monday";
}
else if (x == 3) {
    day = "Tuesday";
}
else if (x == 4) {
    day = "Wednesday";
}
else if (x == 5) {
    day = "Thursday";
}
else if (x == 6) {
    day = "Friday";
}
else if (x == 7) {
    day = "Saturday";
}
else {
    day = "Invalid value";
}

Console.WriteLine("Day: " + day);
```

switch case

```
int x = int.Parse(Console.ReadLine());
string day;

switch (x) {
    case 1:
        day = "Sunday";
        break;
    case 2:
        day = "Monday";
        break;
    case 3:
        day = "Tuesday";
        break;
    case 4:
        day = "Wednesday";
        break;
    case 5:
        day = "Thursday";
        break;
    case 6:
        day = "Friday";
        break;
    case 7:
        day = "Saturday";
        break;
    default:
        day = "Invalid value";
        break;
}

Console.WriteLine("Day: " + day);
```

Expressão condicional ternária

<http://educandoweb.com.br>

Prof. Dr. Nelio Alves


Expressão condicional ternária

Estrutura opcional ao if-else quando se deseja decidir um **VALOR** com base em uma condição.

Sintaxe:

`(condição) ? valor_se_verdadeiro : valor_se_falso`

Exemplos:

`(2 > 4) ? 50 : 80`  **80**

`(10 != 3) ? "Maria" : "Alex"`  **"Maria"**

Demo expressão condicional ternária

```
double preco = 34.5;
double desconto;
if (preco < 20.0) {
    desconto = preco * 0.1;
}
else {
    desconto = preco * 0.05;
}
```

```
double preco = 34.5;
double      = (preco < 20.0) ? preco * 0.1 : preco * 0.05;
```

Aula 88

Funções interessantes para string

<http://educandoweb.com.br>

Prof. Dr. Nelio Alves

Checklist

- Formatar: `ToLower()`, `ToUpper()`, `Trim()`
- Buscar: `IndexOf`, `LastIndexOf`
- Recortar: `Substring(inicio)`, `Substring(inicio, tamanho)`
- Substituir: `Replace(char, char)`, `Replace(string, string)`
- `String.IsNullOrEmpty(str)`, `String.IsNullOrWhiteSpace(str)`
- `str.Split(' ')`
- Conversão para numero: `int x = int.Parse(str)`, `int x = Convert.ToInt32(str)`
- Conversão de número: `str = x.ToString()`, `str = x.ToString("C")`, `str = x.ToString("C3")`, `str = x.ToString("F2")`

string é uma variável do tipo class

```
string original = "abcde FGHIJ ABC abc DEFG ";
string s1 = original.ToUpper();
string s2 = original.ToLower();
string s3 = original.Trim();
int n1 = original.IndexOf("bc");
int n2 = original.LastIndexOf("bc");
string s4 = original.Substring(3);
string s5 = original.Substring(3, 5);
string s6 = original.Replace('a', 'x');
string s7 = original.Replace("abc", "xy");
bool b1 = String.IsNullOrEmpty(original);
bool b2 = String.IsNullOrWhiteSpace(original);

Console.WriteLine("Original: -" + original + "-");
Console.WriteLine("ToUpper: -" + s1 + "-");
Console.WriteLine("ToLower: -" + s2 + "-");
Console.WriteLine("Trim: -" + s3 + "-");
Console.WriteLine("IndexOf('bc'): " + n1);
Console.WriteLine("LastIndexOf('bc'): " + n2);
Console.WriteLine("Substring(3): -" + s4 + "-");
Console.WriteLine("Substring(3, 5): -" + s5 + "-");
Console.WriteLine("Replace('a', 'x'): -" + s6 + "-");
Console.WriteLine("Replace('abc', 'xy'): -" + s7 + "-");
Console.WriteLine("IsNullOrEmpty: " + b1);
Console.WriteLine("IsNullOrWhiteSpace: " + b2);
```

DateTime

<http://educandoweb.com.br>

Prof. Dr. Nelio Alves

DateTime

- Representa um INSTANTE
- É um tipo valor (struct)

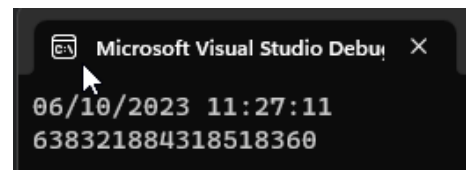
[https://msdn.microsoft.com/en-us/library/system.datetime\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.datetime(v=vs.110).aspx)

- Agenda:
 - Representação interna
 - Instanciação: construtores, builders / conversão String -> DateTime
 - Formatação: DateTime -> String

Representação interna

- Um objeto DateTime internamente armazena:
 - O número de "ticks" (100 nanosegundos) desde a meia noite do dia 1 de janeiro do ano 1 da era comum

```
DateTime d1 = DateTime.Now;  
Console.WriteLine(d1);  
Console.WriteLine(d1.Ticks);
```



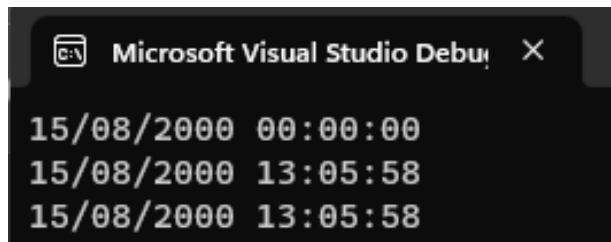
Instanciação

- Construtores
 - DateTime(ano, mes, dia)
 - DateTime(ano, mes, dia, hora, minuto, segundo) *[opcional: kind]*
 - DateTime(ano, mes, dia, hora, minuto, segundo, milissegundo) *[opcional: kind]*
- Builders
 - DateTime.Now
 - DateTime.UtcNow
 - DateTime.Today *[time: 00:00:00]*
 - DateTime.Parse(string)
 - DateTime.ParseExact(string, string)

Demo - construtores

```
DateTime d1 = new DateTime(2000, 8, 15);  
DateTime d2 = new DateTime(2000, 8, 15, 13, 5, 58);  
DateTime d3 = new DateTime(2000, 8, 15, 13, 5, 58, 275);
```

```
Console.WriteLine(d1);  
Console.WriteLine(d2);  
Console.WriteLine(d3);
```

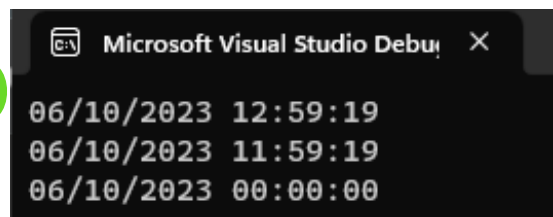


Microsoft Visual Studio Debug Console output:

```
15/08/2000 00:00:00  
15/08/2000 13:05:58  
15/08/2000 13:05:58
```

Demo - Now, UtcNow, Today

```
DateTime d1 = DateTime.Now;  
DateTime d2 = DateTime.UtcNow;  
DateTime d3 = DateTime.Today;  
Console.WriteLine(d1);  
Console.WriteLine(d2);  
Console.WriteLine(d3);
```



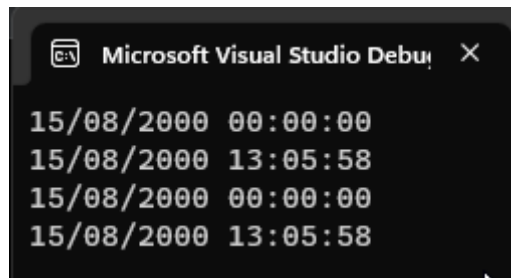
Microsoft Visual Studio Debug Console output:

```
06/10/2023 12:59:19  
06/10/2023 11:59:19  
06/10/2023 00:00:00
```

Demo - Parse

```
DateTime d1 = DateTime.Parse("2000-08-15");  
DateTime d2 = DateTime.Parse("2000-08-15 13:05:58");  
DateTime d3 = DateTime.Parse("15/08/2000");  
DateTime d4 = DateTime.Parse("15/08/2000 13:05:58");
```

```
Console.WriteLine(d1);  
Console.WriteLine(d2);  
Console.WriteLine(d3);  
Console.WriteLine(d4);
```



Microsoft Visual Studio Debug Console output:

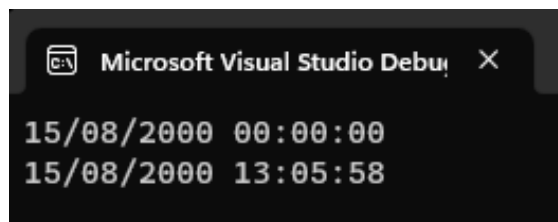
```
15/08/2000 00:00:00  
15/08/2000 13:05:58  
15/08/2000 00:00:00  
15/08/2000 13:05:58
```

Demo - ParseExact

```
DateTime d1 = DateTime.ParseExact("2000-08-15", "yyyy-MM-dd",  
CultureInfo.InvariantCulture);
```

```
DateTime d2 = DateTime.ParseExact("15/08/2000 13:05:58", "dd/MM/yyyy HH:mm:ss",  
CultureInfo.InvariantCulture);
```

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



Microsoft Visual Studio Debug Console output:

```
15/08/2000 00:00:00  
15/08/2000 13:05:58
```

TimeSpan

<http://educandoweb.com.br>

Prof. Dr. Nelio Alves

TimeSpan

- Representa uma DURAÇÃO
- É um tipo valor (struct)

[https://msdn.microsoft.com/en-us/library/system.timespan\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.timespan(v=vs.110).aspx)

Agenda:

- Representação interna
- Instanciação: construtores, fields, métodos From, Parse

Representação interna

- Um objeto `TimeSpan` internamente armazena uma duração na forma de ticks (100 nanosegundos)

```
TimeSpan t1 = new TimeSpan(0, 1, 30);  
Console.WriteLine(t1);  
Console.WriteLine(t1.Ticks);
```



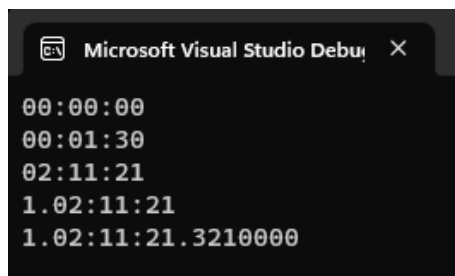
Construtores

- `TimeSpan()`
- `TimeSpan(ticks)`
- `TimeSpan(horas, minutos, segundos)`
- `TimeSpan(dias, horas, minutos, segundos)`
- `TimeSpan(dias, horas, minutos, segundos, milissegundos)`

Demo - construtores

```
TimeSpan t1 = new TimeSpan();  
TimeSpan t2 = new TimeSpan(900000000L);  
TimeSpan t3 = new TimeSpan(2, 11, 21);  
TimeSpan t4 = new TimeSpan(1, 2, 11, 21);  
TimeSpan t5 = new TimeSpan(1, 2, 11, 21, 321);
```

```
Console.WriteLine(t1);  
Console.WriteLine(t2);  
Console.WriteLine(t3);  
Console.WriteLine(t4);  
Console.WriteLine(t5);
```



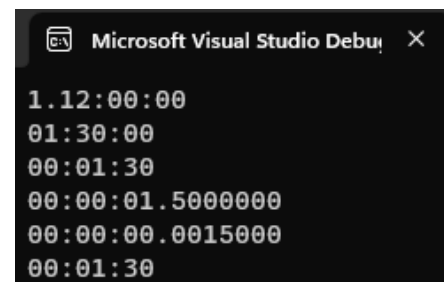
Microsoft Visual Studio Debug Console

```
00:00:00  
00:01:30  
02:11:21  
1.02:11:21  
1.02:11:21.3210000
```

Demo - métodos From

```
TimeSpan t1 = TimeSpan.FromDays(1.5);  
TimeSpan t2 = TimeSpan.FromHours(1.5);  
TimeSpan t3 = TimeSpan.FromMinutes(1.5);  
TimeSpan t4 = TimeSpan.FromSeconds(1.5);  
TimeSpan t5 = TimeSpan.FromMilliseconds(1.5);  
TimeSpan t6 = TimeSpan.FromTicks(900000000L);
```

```
Console.WriteLine(t1);  
Console.WriteLine(t2);  
Console.WriteLine(t3);  
Console.WriteLine(t4);  
Console.WriteLine(t5);  
Console.WriteLine(t6);
```



Microsoft Visual Studio Debug Console

```
1.12:00:00  
01:30:00  
00:01:30  
00:00:01.5000000  
00:00:00.0015000  
00:01:30
```

Propriedades e Operações com DateTime

<http://educandoweb.com.br>

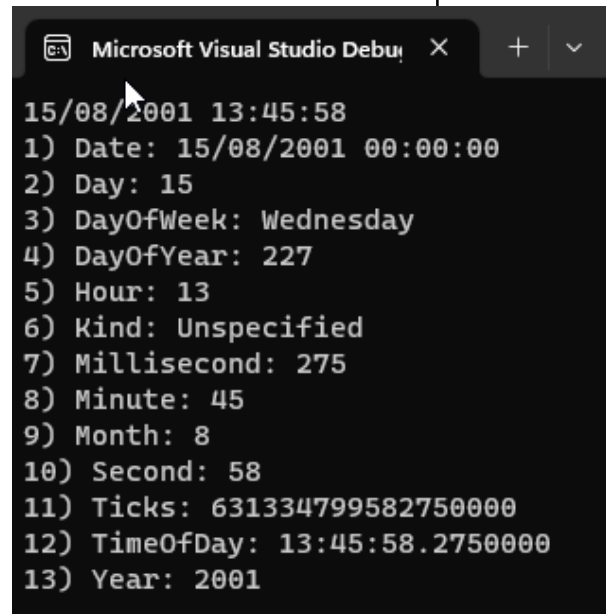
Prof. Dr. Nelio Alves

Propriedades

- Date (DateTime)
- Day (int)
- DayOfWeek (DayOfWeek)
- DayOfYear (int)
- Hour (int)
- Kind (DateTimeKind)
- Millisecond (int)
- Minute (int)
- Month (int)
- Second (int)
- Ticks (long)
- TimeOfDay (TimeSpan)
- Year (int)

Demo

```
DateTime d = new DateTime(2001, 8, 15, 13, 45, 58, 275);
Console.WriteLine(d);
Console.WriteLine("1) Date: " + d.Date);
Console.WriteLine("2) Day: " + d.Day);
Console.WriteLine("3) DayOfWeek: " + d.DayOfWeek);
Console.WriteLine("4) DayOfYear: " + d.DayOfYear);
Console.WriteLine("5) Hour: " + d.Hour);
Console.WriteLine("6) Kind: " + d.Kind);
Console.WriteLine("7) Millisecond: " + d.Millisecond);
Console.WriteLine("8) Minute: " + d.Minute);
Console.WriteLine("9) Month: " + d.Month);
Console.WriteLine("10) Second: " + d.Second);
Console.WriteLine("11) Ticks: " + d.Ticks);
Console.WriteLine("12) TimeOfDay: " + d.TimeOfDay);
Console.WriteLine("13) Year: " + d.Year);
```



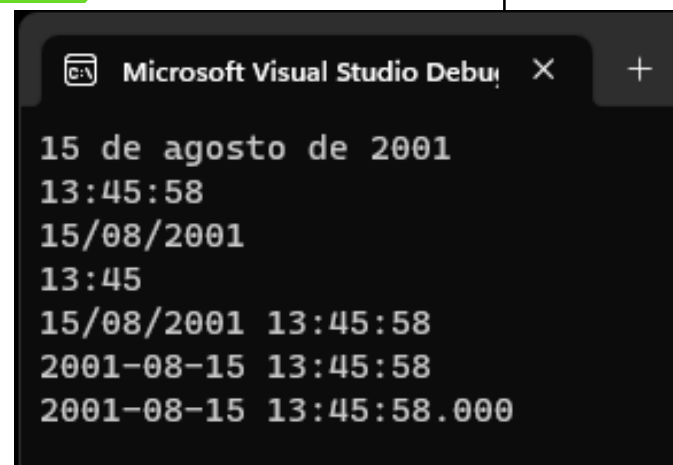
```
Microsoft Visual Studio Debug Console
15/08/2001 13:45:58
1) Date: 15/08/2001 00:00:00
2) Day: 15
3) DayOfWeek: Wednesday
4) DayOfYear: 227
5) Hour: 13
6) Kind: Unspecified
7) Millisecond: 275
8) Minute: 45
9) Month: 8
10) Second: 58
11) Ticks: 631334799582750000
12) TimeOfDay: 13:45:58.2750000
13) Year: 2001
```

Formatação (DateTime -> string)

```
DateTime d = new DateTime(2001, 8, 15, 13, 45, 58);

string s1 = d.ToLongDateString();
string s2 = d.ToLongTimeString();
string s3 = d.ToShortDateString();
string s4 = d.ToShortTimeString();
string s5 = d.ToString();
string s6 = d.ToString("yyyy-MM-dd HH:mm:ss");
string s7 = d.ToString("yyyy-MM-dd HH:mm:ss.fff");

Console.WriteLine(s1);
Console.WriteLine(s2);
Console.WriteLine(s3);
Console.WriteLine(s4);
Console.WriteLine(s5);
Console.WriteLine(s6);
Console.WriteLine(s7);
```



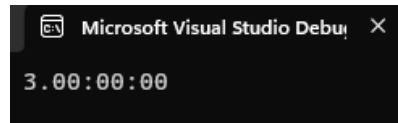
```
Microsoft Visual Studio Debug Console
15 de agosto de 2001
13:45:58
15/08/2001
13:45
15/08/2001 13:45:58
2001-08-15 13:45:58
2001-08-15 13:45:58.000
```

Operações com Datetime

```
DateTime x = ...
```

```
DateTime y = x.Add(TimeSpan);  
DateTime y = x.AddDays(double);  
DateTime y = x.AddHours(double);  
DateTime y = x.AddMilliseconds(double);  
DateTime y = x.AddMinutes(double);  
DateTime y = x.AddMonths(int);  
DateTime y = x.AddSeconds(double);  
DateTime y = x.AddTicks(long);  
DateTime y = x.AddYears(int);
```

```
DateTime y = x.Subtract(TimeSpan);  
TimeSpan t = x.Subtract(DateTime);
```



```
static void Main(string[] args)  
{  
    DateTime x = DateTime.Now;  
  
    DateTime y = x.Add(TimeSpan);  
    Console.WriteLine(y);  
    DateTime y1 = x.AddDays(1);  
    Console.WriteLine(y1);  
    DateTime y2 = x.AddHours(2);  
    Console.WriteLine(y2);  
}
```

```
06/10/2023 14:48:48  
07/10/2023 14:48:48  
06/10/2023 16:48:48
```

Aula 92

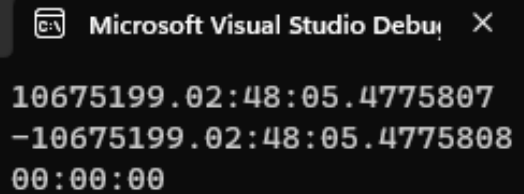
Propriedades e Operações com TimeSpan

<http://educandoweb.com.br>

Prof. Dr. Nelio Alves

Demo: MaxValue, MinValue, Zero

```
TimeSpan t1 = TimeSpan.MaxValue;  
TimeSpan t2 = TimeSpan.MinValue;  
TimeSpan t3 = TimeSpan.Zero;  
Console.WriteLine(t1);  
Console.WriteLine(t2);  
Console.WriteLine(t3);
```

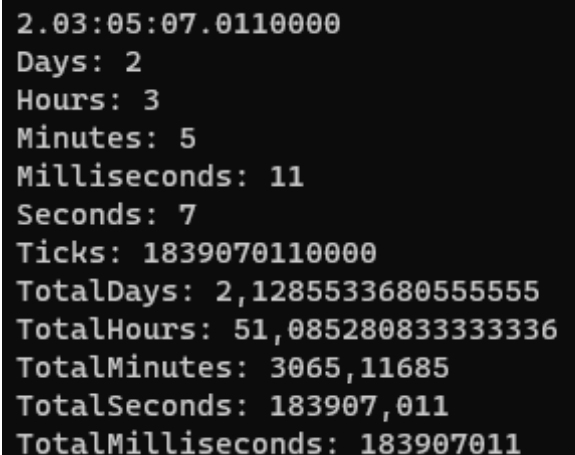


Microsoft Visual Studio Debug Console output:

```
10675199.02:48:05.4775807  
-10675199.02:48:05.4775808  
00:00:00
```

Demo - propiedades

```
TimeSpan t = new TimeSpan(2, 3, 5, 7, 11);  
  
Console.WriteLine(t);  
  
Console.WriteLine("Days: " + t.Days);  
Console.WriteLine("Hours: " + t.Hours);  
Console.WriteLine("Minutes: " + t.Minutes);  
Console.WriteLine("Milliseconds: " + t.Milliseconds);  
Console.WriteLine("Seconds: " + t.Seconds);  
Console.WriteLine("Ticks: " + t.Ticks);  
  
Console.WriteLine("TotalDays: " + t.TotalDays);  
Console.WriteLine("TotalHours: " + t.TotalHours);  
Console.WriteLine("TotalMinutes: " + t.TotalMinutes);  
Console.WriteLine("TotalSeconds: " + t.TotalSeconds);  
Console.WriteLine("TotalMilliseconds: " + t.TotalMilliseconds);
```



Microsoft Visual Studio Debug Console output:

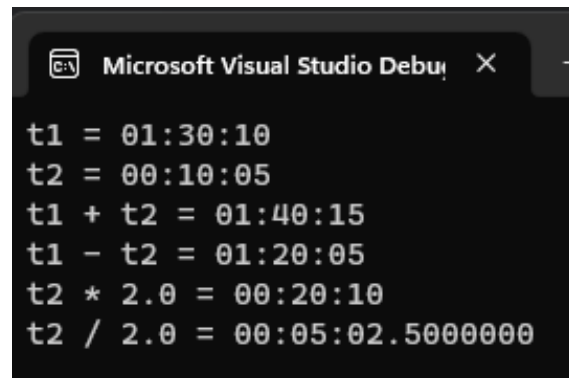
```
2.03:05:07.0110000  
Days: 2  
Hours: 3  
Minutes: 5  
Milliseconds: 11  
Seconds: 7  
Ticks: 1839070110000  
TotalDays: 2,1285533680555555  
TotalHours: 51,085280833333336  
TotalMinutes: 3065,11685  
TotalSeconds: 183907,011  
TotalMilliseconds: 183907011
```

Demo - operações

```
TimeSpan t1 = new TimeSpan(1, 30, 10);  
TimeSpan t2 = new TimeSpan(0, 10, 5);
```

```
TimeSpan sum = t1.Add(t2);  
TimeSpan dif = t1.Subtract(t2);  
TimeSpan mult = t2.Multiply(2.0);  
TimeSpan div = t2.Divide(2.0);
```

```
Console.WriteLine(t1);  
Console.WriteLine(t2);  
Console.WriteLine(sum);  
Console.WriteLine(dif);  
Console.WriteLine(mult);  
Console.WriteLine(div);
```



Microsoft Visual Studio Debug Console output:

```
t1 = 01:30:10  
t2 = 00:10:05  
t1 + t2 = 01:40:15  
t1 - t2 = 01:20:05  
t2 * 2.0 = 00:20:10  
t2 / 2.0 = 00:05:02.5000000
```

Aula 93

DateTimeKind e padrão ISO 8601

<http://educandoweb.com.br>

Prof. Dr. Nelio Alves

DateTimeKind

Tipo enumerado especial que define três valores possíveis para a localidade da data:

- Local *[fuso horário do sistema. Exemplo: São Paulo = GMT -3]*
- Utc *[fuso horário GMT (Greenwich Mean Time)]*
- Unspecified

Boa prática

- Armazenar em formato UTC (texto: BD / Json / XML)
- Instanciar e mostrar em formato Local

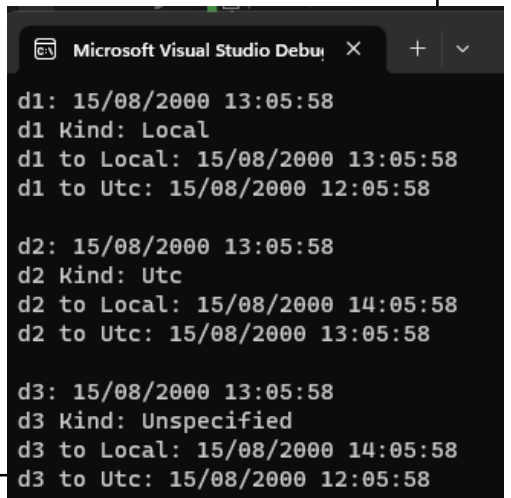
Para converter um DateTime para Local ou Utc, você deve usar:

- myDate.ToLocalTime()
- myDate.ToUniversalTime()

Demo

```
DateTime d1 = new DateTime(2000, 8, 15, 13, 5, 58, DateTimeKind.Local);  
DateTime d2 = new DateTime(2000, 8, 15, 13, 5, 58, DateTimeKind.Utc);  
DateTime d3 = new DateTime(2000, 8, 15, 13, 5, 58);
```

```
Console.WriteLine("d1: " + d1);  
Console.WriteLine("d1 Kind: " + d1.Kind);  
Console.WriteLine("d1 to Local: " + d1.ToLocalTime());  
Console.WriteLine("d1 to Utc: " + d1.ToUniversalTime());  
Console.WriteLine();  
Console.WriteLine("d2: " + d2);  
Console.WriteLine("d2 Kind: " + d2.Kind);  
Console.WriteLine("d2 to Local: " + d2.ToLocalTime());  
Console.WriteLine("d2 to Utc: " + d2.ToUniversalTime());  
Console.WriteLine();  
Console.WriteLine("d3: " + d3);  
Console.WriteLine("d3 Kind: " + d3.Kind);  
Console.WriteLine("d3 to Local: " + d3.ToLocalTime());  
Console.WriteLine("d3 to Utc: " + d3.ToUniversalTime());
```



```
Microsoft Visual Studio Debug Console  
d1: 15/08/2000 13:05:58  
d1 Kind: Local  
d1 to Local: 15/08/2000 13:05:58  
d1 to Utc: 15/08/2000 12:05:58  
  
d2: 15/08/2000 13:05:58  
d2 Kind: Utc  
d2 to Local: 15/08/2000 14:05:58  
d2 to Utc: 15/08/2000 13:05:58  
  
d3: 15/08/2000 13:05:58  
d3 Kind: Unspecified  
d3 to Local: 15/08/2000 14:05:58  
d3 to Utc: 15/08/2000 12:05:58
```

Padrão ISO 8601

- <https://www.iso.org/iso-8601-date-and-time-format.html>
- https://en.wikipedia.org/wiki/ISO_8601

- Formato:

yyyy-MM-ddTHH:mm:ssZ

* Z indica que a data/hora está em Utc

Demo

```
DateTime d1 = DateTime.Parse("2000-08-15 13:05:58");  
DateTime d2 = DateTime.Parse("2000-08-15T13:05:58Z"); // cria local DateTime
```

```
Console.WriteLine("d1: " + d1);  
Console.WriteLine("d1 Kind: " + d1.Kind);  
Console.WriteLine("d1 to Local: " + d1.ToLocalTime());  
Console.WriteLine("d1 to Utc: " + d1.ToUniversalTime());  
Console.WriteLine();
```

```
d1: 15/08/2000 13:05:58  
d1 Kind: Unspecified  
d1 to Local: 15/08/2000 14:05:58  
d1 to Utc: 15/08/2000 12:05:58
```

```
Console.WriteLine("d2: " + d2);  
Console.WriteLine("d2 Kind: " + d2.Kind);  
Console.WriteLine("d2 to Local: " + d2.ToLocalTime());  
Console.WriteLine("d2 to Utc: " + d2.ToUniversalTime());
```

```
d2: 15/08/2000 14:05:58  
d2 Kind: Local  
d2 to Local: 15/08/2000 14:05:58  
d2 to Utc: 15/08/2000 13:05:58
```

```
Console.WriteLine();  
Console.WriteLine(d2.ToString("yyyy-MM-ddTHH:mm:ssZ")); // cuidado!  
Console.WriteLine(d2.ToUniversalTime().ToString("yyyy-MM-ddTHH:mm:ssZ"));
```

```
2000-08-15T14:05:58Z  
2000-08-15T13:05:58Z
```